Wrocław University
of Science and Technology

Doctoral Dissertation

Selected Topics on Randomized Algorithms

Dominik Bojko

Supervisor: prof. dr hab. *Jacek Cichoń*

Branch of science: Mathematics

September, 2021

Acknowledgements

A dissertation in front of you is unduly long and solves a variety of problems. However, I still did not find a solution for one bothering question, namely: "How to beat my supervisor in unconscious hide and seek game?" It is not always easy to find Jacek Cichoń, however, when it finally happens while I have some perplexing issue, he usually simplifies it in some magical way, what usually accelerates my work. I would like to acknowledge my supervisor for this property and for a not to short introduction to a world of science. I would also like to thank my co-authors for teaching me many different styles of work, as well as all my colleagues from the science world, since I can always rely on them and they always guarantee positive aura. I would like to thank for support from Polish National Science Center grant 2018/29/B/ST6/02969 during my work on one of parts of this thesis. I would like to sincerely acknowledge big-hearted Hania Loch-Olszewska for her unexpected move at the beginning of our PhD stories, which made more sense to continue, and probably let me finish this adventure. I cannot forget about my dear friend, Artur Rutkowski, who was from an early age and still is my dispenser of competitive (not only mathematical) motivation. Since we were sucked in a neverending spiral of positive rivalry, we keep it until this day. I think that our PhD studies were reflections of our childhood and I hope that it will be so in the future. I would like to thank my parents for being a balanced support for my whole life. There is no possibility to build a solid building without good foundations. Finally, I would like to admit, that I will not be able to finish my PhD travel without my friends. It definitely helps when we meet more or less regularly and play board games, quizes, drink some beers or simply talk. These meetings let me reset before I can back again to math. I just cannot name them all in here, so I leave a fill-up box for them:
"I would like to thank [ ]." I also really appreciated situations, when someone was bothering me randomly at night just to ask how my PhD thesis is going. It was in fact quite motivating. During PhD period it was not easy to keep a very good physical health and work very effectively at the same time, so I really admired every time I spent on sport with my friends, especially when we were presenting some good football in SZPN. I believe that all the aforementioned issues were intrinsic parts of my PhD story.

Contents

Chapter 1

Introduction

1.1 An outline of the PhD thesis

This doctoral dissertation gathers results from different projects related to probability theory, stochastic processes and algorithmic. It is divided into three main branches: Leader Election Algorithms, Reservoir Sampling Algorithms and Differential Privacy of Probabilistic Counters.

1.1.1 An outline of the Introduction

Chapter 1 is divided in five sections. This Section 1.1 presents the outline of the thesis. Section 1.2 gathers the notation used in this dissertation. We omit the commonly utilized symbols and focus on those not widely used or confusing ones. Section 1.2.1 is devoted to present basic symbols. Further part of Section 1.2 mainly concerns basic facts related to special functions. In Section 1.3, at first we present basic notation, definitions and properties related to probability theory in general. Further we present utilized random variables and theirs variants. In Section 1.4 we recall several standalone theorems used in the thesis and finally in Section 1.5 we shortly present a summary of fundamental concepts needed to understand the whole thesis.

1.1.2 An outline of Leader Election Algorithms part

Chapter 2 is devoted to Leader Election algorithms. Motivation behind this concept and descriptions of possible general arrangements of such the procedures are provided in Section 2.1. A succinct history of Leader Election algorithms is the subject of Section 2.2. In Section 2.3 we describe an idea of Leader Elections via urn model and justify a generality of the procedure in this arrangement.

In Section 2.4 we provide a general block of randomized leader election algorithm consistent with urn model. Especially, we introduce single-hop, multi-hop and beeping models. In Section 2.4.2 we present a general formula for a probability of successful selection of a unique leader by Leader Election in urn model.

Section 2.5 bears the first substantial contribution — for a given number of competitors n and a finite, linearly ordered set A, we attain a distribution of random variable $X(n, A)$, which has the range A and satisfies the following condition: if all the competitors use urn model according to the distribution of $X(n, A)$, then the probability of successful choice of a leader is optimal in a class of Leader Election algorithms designated for n competitors using urn model, where urns are signed by the elements of A. Namely, in Section 2.5.2 and Section 2.5.3, we provide formulas that allow to determine atoms of distributions $X(n, A)$ for different n and A. In Section 2.5.4, we consider a case $n = 2$ separately, for which a distribution of $X(2, A)$ simplifies to uniform one on A. Other fixed cases are considered in Appendix A. Calculation of the probability mass function of the variable $X(n, A)$ for big values of n occurs to be time-consuming, so in Section 2.5.5 we also provide relatively precise approximations for atoms of distribution of $X(n, A)$. They are utilized in a proposition of efficient, easy and quick to calculate approximations of the optimal distributions. This part is postponed until Appendix A, where we provide comparisons of optimal solution and the approximated ones via numerical calculations. In the last Section 2.5.6, we provide an astonishing fact about the necessary and sufficient memory supplies (and simultaneously bounds on the runtime) to guarantee reliable non-anonymous Leader Election in urn model.

A next Section 2.6 is a foundation of the rest of the chapter. It introduces a very simple quasi-anonymous Uniform Leader Election algorithm, consistent with randomized urn model, realized according to the uniform distribution. We show that similar results were considered before, however in slightly different language. Further we provide a novel, yet very natural result, showing the monotonicity of the success probability with respect to the number of competitors. Further we provide several remarks related to Uniform Leader Election. In Section 2.6.4 we provide a general and optimal lower bound on the time complexity of quasi-anonymous Leader Election algorithms and we compare it with results for Uniform Leader Election and the similar result from Section 2.5.6 for non-anonymous algorithms.[1]

Further, in Section 2.7, we describe previously known results closely related to our contribution. In particular, the one considered to be the present state-of-the-art of universal Leader Election algorithms. Moreover, we briefly analyze accuracy of these procedures and theirs alternative versions, proposed in this section.

Next Section 2.8 generalizes the concept of quasi-anonymous Leader Green Election algorithm according to geometric distribution, which was proposed by P.Jacquet [64]. We refine some crucial ideas of P.Jacquet and produce more flexible solution, called Geometric Green Leader Election (shortly GeoGLE). Our procedure utilizes restricted version of geometric distribution with precisely tuned parameters. In Section 2.8.2 we present GeoGLE algorithm in the single-hop arrangement of urn model and further we provide Theorem 2.8.1, which appoints the parameters of GeoGLE algorithm, for a given minimal probabil-

[1] To the best of our knowledge, such the bound was known before only for specific solutions.

ity of a successful Leader Election and a constraint on the maximal number of competitors. A technical proof of Theorem 2.8.1 is postponed to Appendix B. A discussion about GeoGLE solution and its effectiveness are included in Section 2.8.3. An implementation's details are the topic of Section 2.8.4.

The latter crucial contribution in Chapter 2 is described in Section 2.9. It combines the advantages of Uniform Leader Election and GeoGLE algorithms from Section 2.6 and Section 2.8 by juxtaposition of both procedures. We show that our solution is almost as good in terms of a total runtime as GeoGLE algorithm and it can be still rectified. A part of calculations are postponed to Appendix C. A summary of all the considered Leader Election algorithms is provided in Section 2.10.

1.1.3 An outline of Reservoir Sampling Algorithms part

Next Chapter 3 is devoted to a particular domain of Big Data analysis, namely to Reservoir Sampling algorithms. Section 3.1 describes the essential ideas behind this kind of procedures, including theirs short history, starting from the famous Algorithm R. In Section 3.2 we introduce a concept of sliding window algorithms and provide a general Reservoir Sampling algorithm in a sliding window of the fixed discrete size assumed a priori. A fundamental algorithm bases on devil's staircase Markov chains, described in Section 3.2.4. After we present the basic Reservoir sampling algorithm and the constraint of its applicability in Section 3.2.5, we present several artful tricks, which improve the efficiency of our solution and next we show how to extend our algorithm to a case of uniform sampling in Section 3.2.7. In further sections we generalize the extension to a wider class of distributions. In Section 3.2.8 we analyze several examples of possible distributions of samples in the sliding window together with theirs properties. We end up Section 3.2 with a description of admissible class of distributions, that can be modelled thanks to our contribution and present an exemplary application.

In Section 3.3 we provide a Reservoir Sampling algorithm with update probabilities taken according to a Power Law of Update which depends on the number of data items already explored. An idea and general algorithm are presented in Section 3.3.1. Section 3.3.2 itemizes diverse laws of convergence, together with asymptotical expected values of the pointers of the data sampled by our algorithm, with respect to cases dependent on the definitions of update probabilities. Further, Section 3.3.3 contains general properties of Power Law of Update model of Reservoir Sampling algorithms. Next 6 consecutive subsections: 3.3.4, 3.3.5, 3.3.6, 3.3.7, 3.3.8 and 3.3.9 provide the analysis for each of the cases specified in Section 3.3.2, dependent on the exponent of the Power Law of Update. In Section 3.3.10 we present several ideas of applications of Power Law of Update solution. Appendix D provides more precise results for some particular cases of Power Law of Update model with rational exponents.

1.1.4 An outline of Differential Privacy of Probabilistic Counters part

The last Chapter 4 is devoted to differential privacy of probabilistic counters. Section 4.1 provides basic intuitions beneath notions of differential privacy property and a motivation to analyze probabilistic counters in terms of this attribute. We evoke standard examples of Morris and MaxGeo counters and theirs potential applications. The rest of chapter Chapter 4 is organized as follows. Firstly, in Section 4.2 we recall differential privacy definition. In Section 4.3 we define both Morris and MaxGeo counters and state Fact 4.3.1, which is a useful reformulation of the standard differential privacy definition for probabilistic counters.[2] It worth to emphasize, that the concept of probabilistic counter is ambiguous in literature and we provide its formal, general definition in Section 1.3.4. Formulation of our most significant technical contribution is presented in Section 4.5, where we analyse how both aforementioned counters behave under differential privacy regime. Namely, we prove that under certain assumptions about the minimal number of requests, both probabilistic counters are differentially private by design, with some parameters, which tends to 0, as the number of users grow.[3] For the sake of clarity some technical proofs and lemmas are moved to Appendix E. It worth to note, that the authors of [33] showed that probabilistic counters do not preserve differential privacy when they are utilized to estimate a set cardinality with a strict constraint that the adversary is able to extract the intermediate values of counter. We attain the opposite result by easing this restriction. We also recall some applications of MaxGeo Counter, which, thanks to our contribution, can be trivially transformed to differentially private mechanisms. In Section 4.6 we demonstrate how probabilistic counters can be used for constructing a data aggregation protocol in a very particular, yet natural, scenario. We also provide several possible directions of applications of our results. Further, we compare these protocols with state-of-the-art Laplace method based solution, which provides differential private data aggregation mechanism using extra randomization. In Section 4.7 we recall work related to main topics of Chapter 4 and some popular examples of other probabilistic counters, which are not considered in this chapter. Finally, in Section 4.8 we summarize our contribution, present conclusions and several future work propositions.

1.1.5 Rest of the dissertation

After appendices, we placed lists of all tables, figures and algorithms present in the thesis for a convenient navigation. A bibliography closes this dissertation.

[2] In fact it provides a convenient sufficient condition that guarantees differential privacy of probabilistic counters.

[3] The closer the parameters are to 0, the more private is the mechanism.

1.2 Notations and an overview of special functions

1.2.1 Symbols

First we want to present some symbols, used in the dissertation, which can be unknown or confusing. When we want to define some symbol s via formula F, then we depict it as $s := F$.[4] We denote the cardinality of a set A by $\operatorname{card}(A)$.

In order to not settle the dispute if 0 is a natural number or not, we disambiguate the notation by using two different symbols: $\mathbb{N}$ – which is a set of positive integers – and $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$ – a set of non-negative integers. For $a, b \in \mathbb{Z}$, $a \leqslant b$, we define a discrete interval $[a : b] := [a, b] \cap \mathbb{Z}$. If $a > b$, then $[a : b] = \emptyset$. We eventually allow b to be ∞. Then $[a : b]$ is a set of all integers, greater or equal to a. Especially, for $n \in \mathbb{N}$, we define $[n] := [1 : n]$. It is useful to slightly extend this definition, so we denote $[\infty] = \mathbb{N}$.
A set of all integers, rational, real and complex numbers are denoted respectively as $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$ and $\mathbb{C}$. $\Re z$ denotes the real part of a complex number z.
A half of the length of the circumference of a unit circle is denoted by π.

For $x \in \mathbb{R}$, we define a fractional part of x, i.e. $\{x\}_f := x - \lfloor x \rfloor$.[5] Rounding a number x is given as $\lfloor x \rceil$ and it is defined as the closest integer to x. When $2x$ is odd integer, then $\lfloor x \rceil = x + \frac{1}{2}$.

By $(c_n)_{n=a}^{b}$ we indicate a sequence c defined on $[a : b]$. When sometimes the domain do not need a particular attention, we omit it and write simply $(c_n)_n$.[6] When an element of a sequence c_n emerges in pseudo-code, when we denote it by $c[n]$ instead. A concatenation of two sequences $c = (c_n)_{n=1}^{k}$ and $d = (d_n)_{n=1}^{l}$ is given as

$$(c \| d)_n := \begin{cases} c_n & \text{, when } n \in [k] \text{ ,} \\ d_{n-k} & \text{, when } n \in [k+1 : k+l] \text{ .} \end{cases}$$ [7]

Alternatively, if $d = (d_n)_{n=k+1}^{k+l}$, then we simply define concatenation of sequences $(c)_n$ and $(d)_n$ by $(c \cup d)_n$.

We define horizontal vectors similarly to sequences. Indeed, if $v \in V^n$, then we denote them as $[v_i]_{i=1}^{n}$, where all $v_i \in V$. Another matrices will be defined, when needed.

Intuitively, the concatenation is just glueing the second sequence after the end of the first one.

---

[4] := notation is utilized in Pascal programming language to assign data to variables. When we want to assign the left hand side to the right one, we use =: instead.

[5] Note that in the literature, the fractional part of x is usually denoted by $\{x\}$. However we add "f" in the subscript in order to distinguish this notation from the one of a singleton of element x.

[6] Sometimes it is convenient to write the sequence in the reversed order. We indicate this by swapping the positions of a and b (when $a \leqslant b$).

[7] We assume that $k \in \mathbb{N}$ and $l \in \mathbb{N} \cup \{\infty\}$ with a notational convention that $k + \infty = \infty$.

Indicator of a set A is a function defined as:

$$\mathbf{1}_A(x) := \begin{cases} 1 & \text{, when } x \in A \text{ ,} \\ 0 & \text{, when } x \notin A \text{ .} \end{cases}$$

If E is a statement, then an Iverson bracket of E is defined as a function of the free variables in E:

$$[\![E]\!] = \begin{cases} 1 & \text{, if } E \text{ is true ,} \\ 0 & \text{, if } E \text{ is false .} \end{cases}$$

The Iverson bracket is a generalization of both Kronecker's δ and indicator notation. Indeed, $[\![x = y]\!] = \delta_{xy}$ and $[\![x \in A]\!] = \mathbf{1}_A(x)$.

Let $a \in \mathbb{R}$ and f be a function defined on a subset of $\mathbb{R}$, continuous in some vicinity of a (possibly excluding a). Then by $f(x)|_{x=a}$ we denote either evaluation of f at a (i.e. $f(a)$) or a limit as x tends to a (i.e. $\lim\limits_{x \to a} f(x)$). When the limit does not exist, but there exist one-side limits, then we write $f(x)|_{x=a^+}$ or $f(x)|_{x=a^-}$, respectively, for right-side or left-side limit.

We use uniqueness quantification, when we want to pay attention that exactly one of the terms $E(x)$ is fulfilled by the non-free variable x. We indicate it by addition of '!' sign: $(\exists!\, x)E(x)$.

If $a, b \in \mathbb{Z}$, then $\mathrm{GCD}(a, b)$ depicts the greatest common divisor of $|a|$ and $|b|$. Let $a \in \mathbb{R}$ and $b > 0$. Then we define $a \bmod b$ as $c \in [0, b)$, such that $a + kb = c$, for some $k \in \mathbb{Z}$. Then $a \div b = \frac{a - a \bmod b}{b}$.[8]

Where we want to indicate that a function f constantly equals c, then we write $f \equiv c$.

1.2.2 Norm and scalar product

Let $n \in \mathbb{N}$ and $x, y \in \mathbb{R}^n$. We are going to utilize two standard *norms* on spaces of finite sequences:*For general definitions see any coursebook on functional analysis.*

- ℓ_1-norm, defined as

$$\|x - y\|_1 = \sum_{i \in [n]} |x_i - y_i| \text{ ,}$$

- ℓ_2-norm, defined as

$$\|x - y\|_2 = \sqrt{\sum_{i \in [n]} (x_i - y_i)^2} \text{ .}$$

[8] Usually mod and $\div$ are used for $a > 0$ and $b \in \mathbb{N}$.

Moreover, a *scalar product* is defined as

$$\langle x, y\rangle = \sum_{i\in[n]} x_i y_i \ .$$

Apart from them, we are going to define another one, called total variation distance (defined in Section 1.3.13), which is closely related to ℓ_1 norm.

1.2.3 Order relation

If $\preceq \subset X \times X$ is reflexive ($x \preceq x$), antisymmetric ($((x \preceq y) \wedge (y \preceq x)) \Rightarrow x = y$) and transitive ($((x \preceq y) \wedge (y \preceq z)) \Rightarrow (x \preceq z)$) relation,[9] then we call it a *partial order* on X. If the partial order $\preceq$ satisfies $(\forall\, x, y \in X)\ x \preceq y \vee y \preceq x$, then it is a *linear order* on X. If the linear order $\preceq$ has the property that for each $A \subset X$, there exists $a \in A$ such that $(\forall\, x \in A\backslash\{a\})\ a \preceq x$, then we call $\preceq$ a *well order*. If $\preceq$ is the partial order, then $\preceq^*$ defined as $\{(x, y) \in X \times X : y \preceq x\}$ is a *reversed order* with respect to $\preceq$. If an order $\preceq$ is reversed to some well order $\preceq^*$, then one can define $\max_{\preceq}(A)$, as the biggest element of the subset A, i.e. the smallest element this set with respect to $\preceq^*$ order. If $\preceq$ is a partial order on X and $\leqslant$ is a partial order on Y, then one can define a *lexicographic order* on $X \times Y$ by the formula: $\{((x_1, y_1), (x_2, y_2)) \in (X \times Y)^2 : x_1 \preceq x_2 \vee ((x_1 = x_2) \wedge (y_1 \leqslant y_2))\}$.

Fact 1.2.1. *Every partial order can be extended to some linear order.*

1.2.4 Sums, products, derivatives and integrals

We use standard notation for sums, products and integrals. However, for convenience, we highlight the convention of the limits of these operations. Indeed, if $a, b \in \mathbb{N}$ and $b < a$, then $\sum_{n=a}^{b} c_n = 0$ and $\prod_{n=a}^{b} c_n = 1$, since they are empty operations. Nevertheless, if $a, b \in \mathbb{R}$ and $b < a$, then naturally $\int_a^b c(x)dx = -\int_b^a c(x)dx$.

Fact 1.2.2. *If $(a_0, a_1, \ldots, a_n)$ is a sequence of real numbers, then*

$$\sum_{k=1}^{n} k(a_k - a_{k-1}) = na_n - \sum_{k=0}^{n-1} a_k \ . \tag{1.1}$$

A discrete simplex of k-parts compositions of integers from $[n]$ (see e.g. [47] for definition of the composition of integer) is defined as

$$\mathrm{Sim}_n^{(k)} := \left\{ \bar{l} = [l_i]_{i=1}^k \in \mathbb{N}_0^k : \sum_{i=1}^{k} l_i \leqslant n \right\} \ .$$

[9] All conditions are for all $x, y, z \in X$.

Derivatives of product of functions

Usually we denote a derivative of function f by f'. If a function f has more than one variable or uses some parameters, then when it is not clear from the context, we write $\frac{\partial f}{\partial x}$ to denote that we calculate the derivative of f with respect to x variable (or eventually a parameter).

If functions f, g are n-times differentiable, then $f^{(n)}$ denotes the n-th derivative of f and

$$(f \cdot g)^{(n)}(x) = \sum_{k=0}^{n} \binom{n}{k} f^{(k)}(x) g^{(n-k)}(x) \ . \tag{1.2}$$

Theorem 1.2.1 (Mean Value Theorem)**.** *If $f : [a, b] \to \mathbb{R}$ has continuous derivative, then*

$$f(b) - f(a) = f'(c)(b - a) \ ,$$

for some $c \in [a, b]$.

Series forms of functions

We say that f has

- an expansion to a Taylor series at z_0, when:

$$f(z) = \sum_{n=0}^{\infty} a_n (z - z_0)^n \ ,^{10}$$

- an expansion to a Laurent series at z_0, when:

$$f(z) = \sum_{n=-\infty}^{\infty} a_n (z - z_0)^n \ .$$

Let $r \in \mathbb{R}$. By $[(z - z_0)^r] f(z)$ we understand an extraction of a coefficient corresponding to $(z - z_0)^r$ in the formal power series of $f(z)$.[11] We call it a *$(z - z_0)$-term extractor*.[12] When we deal with Taylor or Laurent series, then r are respectively from $\mathbb{N}_0$ and $\mathbb{Z}$. However, in Appendix D we will also consider $r \in \mathbb{Q}$ case, i.e. series with rational powers.

Integral approximation of a sum

If f is a non-decreasing, integrable function on A and $[a - 1 : b + 1] \subset A$, then

$$\int_{a-1}^{b} f(x)dx \leqslant \sum_{n=a}^{b} f(n) \leqslant \int_{a}^{b+1} f(x)dx \ . \tag{1.3}$$

[10] When $z_0 = 0$, then we obtain Maclaurin series.

[11] Formal power series is an algebraic interpretation of the notion of series of powers of some variable (z in this case). When a power series is convergent in some region, then we may consider it analytically in this set.

[12] Especially, when $z_0 = 0$, we call it z-term extractor.

Convergence tests

Consider a sequence $a := (a_i)_{i=0}^{\infty}$ and its corresponding ordinary generating function[13]:

$$A_a(z) := \sum_{i=0}^{\infty} a_i z^i \ ,$$

we define a radius of a series A_a as $R(a) := \left(\limsup_{n\to\infty} \sqrt[n]{|a_n|} \right)^{-1}$.

Fact 1.2.3 (Cauchy root test)**.**

- *$A_a(z)$ is point-wise convergent in a disk $\{z \in \mathbb{C} : R(a) > |z|\}$,*
- *If $R(a) > |z|$, then $A_a(z)$ diverges.*

We call a sequence $a \in \mathbb{R}^{\mathbb{N}}$ an alternating, if all terms $(-1)^n a_n$ have the same sign. If a is alternating and a sequence $(|a_n|)_n$ converges to 0 monotonically, then a is a Leibniz's type sequence.

Fact 1.2.4 (Leibniz's test)**.** *If a is Leibniz's type, then*

$$A_a(1) = \sum_{i=1}^{\infty} a_n$$ [14]

is convergent and each consecutive partial sum provides more precise approximation of $A_a(1)$. Moreover, for any two consecutive partial sums, one is bigger than $A_a(1)$ and the second is smaller than $A_a(1)$.

1.2.5 Asymptotics: Bachmann—Landau and Vinogradov notations

We would like to introduce few asymptotics notations. First are due to Bachmann and Landau and are gathered in a family of so called Bachmann—Landau notation family. It worth to mention, that the presently used notation is due to Knuth (see the book of Knuth et al. [54] for the history of the notation and wider description).

A class of all open sets which contains a point x, we denote by $\mathcal{B}(x)$.

Let f be a real (or complex) function and g be a positive function. Moreover, let f and g have a common domain, which contains some $A \in \mathcal{B}(x_0)$.

In Table 1.1, we present a definitions of different Bachmann—Landau symbols. We also allow x_0 to be $\pm\infty$, which is a typical case of application. We sometimes write equations of a form $f(x) = h(x) + \mathrm{O}\,(g(x))$, what is equivalent to $f(x) - h(x) = \mathrm{O}\,(g(x))$,[15] where $\mathrm{O}\,()$ notation can be substituted with

[13] Ordinary generating function is sometimes called ordinary formal power series. The attribute "ordinary" is due the simple form of the i-th coefficient — a_i.

[14] This type of series we also call a Leibniz's type series.

[15] One sometimes write $f(x) - h(x) \in \mathrm{O}\,(g(x))$ to indicate that $\mathrm{O}\,(g(x))$ is a class of functions.

Notation	Definition
$f(x) = \mathrm{O}\,(g(x))$	if $(\exists\, M > 0)(\exists\, A \in \mathcal{B}(x))(\forall\, x \in A)\ \lvert f(x)\rvert \leqslant Mg(x)$
$f(x) = \mathrm{o}\,(g(x))$	if $(\forall\, \varepsilon > 0)(\exists\, A \in \mathcal{B}(x))(\forall\, x \in A)\ \lvert f(x)\rvert < \varepsilon g(x)$
$f(x) = \Omega\,(g(x))$	if $(\exists\, \varepsilon > 0)(\exists\, A \in \mathcal{B}(x))(\forall\, x \in A)\ \lvert f(x)\rvert \geqslant \varepsilon g(x)$
$f(x) = \omega\,(g(x))$	if $(\forall\, M > 0)(\exists\, A \in \mathcal{B}(x))(\forall\, x \in A)\ \lvert f(x)\rvert > Mg(x)$
$f(x) = \Theta(g(x))$	if $f(x) = \mathrm{O}\,(g(x))$ and $f(x) = \Omega\,(g(x))$
$f(x) \sim g(x)$	if $(\forall\, \varepsilon > 0)(\exists\, A \in \mathcal{B}(x))(\forall\, x \in A)\ \left\lvert \frac{f(x)}{g(x)} - 1\right\rvert \leqslant \varepsilon$

Table 1.1: Definitions of several Bachmann—Landau symbols, as $x \to x_0$.

other Bachmann—Landau notations (excluding $\sim$). We also allow to write these asymptotics notations on both sides of equations if there exists a function, which can be satisfied by both conditions.

In 1930s, Vinogradov introduced another, convenient alternative notation for $\mathrm{O}\,()$: $f \ll g \Leftrightarrow f(x) = \mathrm{O}\,(g(x))$.[16]

One can also define similar multivariable notation. It can be naturally generalized by defining open sets as products of open sets over disjoint coordinates.

1.2.6 Exponents and logarithms

A Neper—Euler constant e is defined as the limit $\lim\limits_{n\to\infty} \left(1 + \frac{1}{n}\right)^n$.
An exponential function with base e, is written either as e^x or $\exp(x)$ and is given by Maclaurin series:

$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!}\ . \tag{1.4}$$

A natural logarithm (of base e) is denoted by $\ln(x)$ and its Taylor series at $x = 1$ can be rewritten as

$$\ln(1+x) = \sum_{n=1}^{\infty} \frac{(-1)^{n-1} x^n}{n}\ . \tag{1.5}$$

Directly from the above we may obtain that for $x \in (0, 1)$,

$$-x\left(1 + \frac{x}{2(1-x)}\right) < \ln(1-x) < -x\left(1 + \frac{x}{2}\right)\ . \tag{1.6}$$

By $\lg(x)$ we denote a binary logarithm (of base 2). An iterated logarithm (also known as log-star), denoted by $\lg^*(x)$ is a function defined as $\min\{i \in \mathbb{N} : \lg^{[i]}(x) \leqslant 2\}$, where $\lg^{[1]}(x) = \lg(x)$ and $\lg^{[i+1]}(x) = \lg(\lg^{[i]}(x))$, for $i \in \mathbb{N}$. The iterated logarithm is the function, which intuitively ascends very slowly. There

[16] In fact, this notation is antisymmetric version of du Bois-Reymond's notation from 1871. See e.g. [54] for a wider context.

N	1	4	5	16	17	2^{16}	$2^{16}+1$	$2^{2^{16}}$	$2^{2^{16}}+1$
$\lg^* N$	1	1	2	2	3	3	4	4	5

Table 1.2: Several crucial values of $\lg^*$ function.

is one function commonly used in computability theory, called inverse Ackermann's function, which is rising slower than the iterated logarithm in terms of Bachmann—Landau notation (see e.g. [27] for more details). In Table 1.2, we present some values of $\lg^*$ function, which indicates the arguments at which $\lg^*$ is incremented. According to [92], the number $2^{2^{16}}$ has 19 729 digits.[17] In this paper, we will only consider integer arguments of the above function.

1.2.7 Binary representations of numbers and messages

Natural languages are not intelligible for computers per se. However any message can be translated for example, to some binary representation, which in digitalized form can be read by a machine. If this device know the arrangement of the translation, then the message can be "understood" by the computer. A binary representation of a message msg is a finite sequence (b_i) of numbers from the set $\{0, 1\}$. Usually messages are meant to be just some sentences in some natural languages or some communicates. Such the symbols can be written, for instance, in ASCII or Unicode, which are naturally translatable either to natural numbers or to binary representations. Therefore sentences or communicates can be easily translated to sequences of zeros and ones. Moreover, often messages are simultaneously interpreted as some big natural numbers. For the sake of clarity we assume that all the messages are simply the non-negative integers. By BIN(msg) we denote the binary representation of msg $\in \mathbb{N}_0$ message. Precisely, if $b = \mathrm{BIN}(\mathrm{msg})$ and $\lceil \lg(\mathrm{msg}+1) \rceil = K$, then b consists of K bits, numbered in inverted order from $K-1$ down to 0, i.e. $b = (b_i)_{i=K-1}^{0}$. We assume that $\mathrm{BIN}(0) = (0)$ and $\mathrm{BIN}(1) = (1)$. Moreover, for $\mathrm{msg} \leqslant 2$ and $s = \mathrm{msg} \bmod 2$, we can define b recursively as a concatenation of $\mathrm{BIN}(\frac{\mathrm{msg}-s}{2})$ and (s). Then naturally $b_0 = \mathrm{msg} \bmod 2$ and

$$\mathrm{msg} = \sum_{i=0}^{K-1} b_i 2^i \ .$$

For example $\mathrm{BIN}(11) = (1, 0, 1, 1)$ and $11 = 1\cdot 2^0 + 1\cdot 2^1 + 0\cdot 2^2 + 1\cdot 2^3$. Moreover, for a given $K \in \mathbb{N}$, we can define a mapping $\mathrm{BIN}_K(\mathrm{msg})$ for $\mathrm{msg} \in [0 : 2^K - 1]$. Indeed, we put a concatenation of a sequence consisting of $k := K - \lceil \lg(\mathrm{msg} + 1) \rceil$ zeros and BIN(msg). This way we always get a binary representation of length K.

[17] One can recognize that $2^{2^{16}} = 2 \uparrow\uparrow 5$ in terms of Knuth's up-arrow notation (see [70] for definition). One can realize that the inverse of the function $2 \uparrow\uparrow x$ is closely related to $\lg^* x$.

1.2.8 Factorials, Euler's Gamma and Binomials

Factorials

When $n \in \mathbb{N}_0$, then $n!$ is n-factorial defined as $\prod_{k=1}^{n} k$.

Let $r \in \mathbb{C}$ and $s \in \mathbb{N}$. A s-th falling factorial of r is defined as

$$[r]_s := \prod_{i=0}^{s-1} (r-i) \ .$$

Euler's Gamma and Beta Functions

When $z \in \mathbb{C}$ and $\Re(z) > 0$, then we can define Gamma function $\Gamma(z) = \int_0^\infty t^{z-1} e^{-t} dt$. For $z \in \mathbb{N}$, $\Gamma(z) = (z-1)!$. Moreover Gamma function can be extended to $\mathbb{C} \backslash \{-n : n \in \mathbb{N}_0\}$ in such the way that

Fact 1.2.5. $z\Gamma(z) = \Gamma(z+1)$.

We define $\mathrm{B}(a,b) := \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}$, whenever the right hand side makes sense. Alternatively, for $a, b > 0$, we can define it by the formula $\mathrm{B}(a,b) = \int_0^1 t^{a-1}(1-t)^{b-1} dt$.

Binomial coefficients

Let $n, k \in \mathbb{N}_0$ and $n \geqslant k$. Then we define Newton's binomial $\binom{n}{k}$ as $\frac{n!}{k!(n-k)!}$.

When $r \in \mathbb{C}$ and $k \in \mathbb{N}$, we can define Newton's binomial by $\binom{r}{k} = \frac{[r]_k}{k!}$.

Fact 1.2.6 (Newton's Binomial Formula)**.** *If $n \in \mathbb{N}$ and $|x| < 1$, then*

$$(1+x)^n = \sum_{k=0}^{n} \binom{n}{k} x^k \ .$$

If $r \notin \mathbb{N}$ and $|x| < 1$, then

$$(1+x)^r = \sum_{k=0}^{\infty} \binom{r}{k} x^k \ .$$

Fact 1.2.7 (From [108])**.** *Let $z \in \mathbb{C}$ and $n \in \mathbb{N}_0$ such that $\arg(n-z) \neq \pi$ (i.e. $n - z$ is not a negative real number), then*

$$\binom{z}{n} = \frac{\Gamma(z+1)\sin(\pi(n-z))}{\pi k^{z+1}} \left(1 + \frac{(z+1)z}{2n} + \mathrm{O}\left(n^{-2}\right)\right) \ ,$$

as $n \to \infty$.

1.2.9 Harmonic numbers

H_n is n-th harmonic number, defined as $H_n = \sum_{i=1}^{n} \frac{1}{i}$.

${H_n}^{(k)} = \sum_{i=1}^{n} \frac{1}{i^k}$ is the generalized n-th harmonic number of the k-th kind. In particular ${H_n}^{(1)} = H_n$. Usually we consider only $k \geqslant 1$.

Fact 1.2.8. *An Euler—Mascheroni constant $\gamma = 0.577215\ldots$ is a limit of a sequence $(H_n - \ln n)_{n=1}^{\infty}$.*

Fact 1.2.9. *An asymptotics of harmonic numbers when $n \to \infty$ is as follows:*

$$H_n = \ln n + \gamma + \frac{1}{2n} - \frac{1}{12n^2} + \mathrm{O}\left(n^{-4}\right) \ . \tag{1.7}$$

Next two facts can be obtained e.g. via Euler—Maclaurin summation formula (1.14):

Fact 1.2.10. *If $a > 1$, then the following formula is useful:*

$${H_n}^{(a)} - {H_k}^{(a)} = \frac{k^{1-a} - n^{1-a}}{a-1} + f_a(k) + \mathrm{O}\left(1\right) \ ,$$

as $n \to \infty$, where $|f_a(k)| \leqslant k^{-a}$.

Fact 1.2.11. *When $a = 1$, we deal with the standard harmonic numbers and achieve the following approximation:*

$$H_n - H_k = \ln(n) - \ln(k) + f_1(k) + \mathrm{O}\left(n^{-1}\right) \ ,$$

as $n \to \infty$, where $|f_1(k)| \leqslant k^{-1}$.

1.2.10 Riemann's Zeta function

When a real part of complex number z satisfies $\Re z > 1$, we can define Riemann's Zeta function for such the argument as:

$$\zeta(z) = \sum_{n=1}^{\infty} \frac{1}{n^z} \ .$$

Fact 1.2.12. *For $k > 1$, $\lim_{n\to\infty} {H_n}^{(k)} = \zeta(k)$.*

Fact 1.2.13. *$\zeta(2) = \frac{\pi^2}{6}$ and $\zeta(3) = 1.2020569\ldots < \frac{\pi^3}{25}$.*

1.2.11 PolyLogarithms

Polylogarithm (or Jonquiére's function) $\mathrm{Li}_k\left(x\right)$ is defined as $\sum_{n=1}^{\infty} \frac{x^n}{n^k}$, whenever the series converges.[18] A name of polylogarithm can be partially justified by the short form of $\mathrm{Li}_1\left(x\right)$, which is a Maclaurin expansion of a function $-\ln(1-x)$.

[18] Do not confuse polylogarithm with polylogarythmic function, which is a polynomial in the logarithm of a variable.

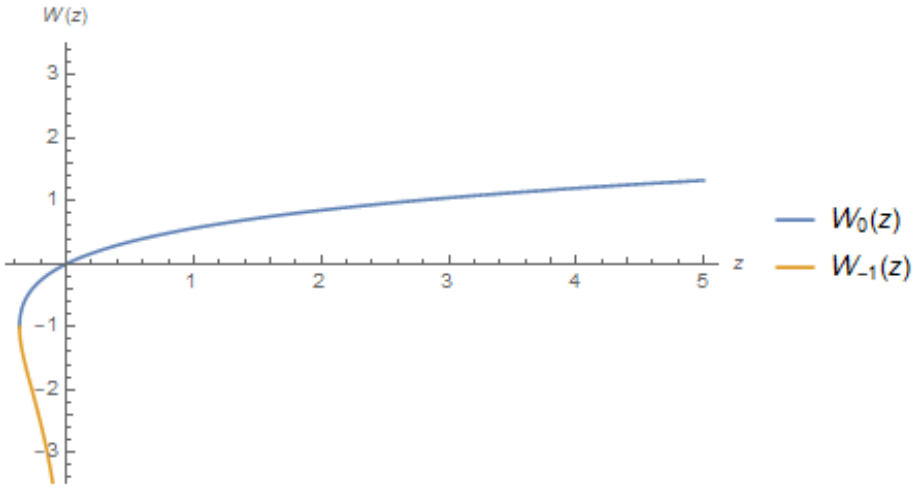

Figure 1.1: Plots of real values of two basic branches of W-Lambert function.

1.2.12 W-Lambert multi-function

W-Lambert function is defined as an inverse of the complex function $f(x) = xe^x$, i.e. $y = W(x)$ if $x = ye^y$. In fact it is a multi-function, because f is not injective and in consequence W can take more than one value at some points. In order to disambiguate the values, the multifunction is divided into separate functions, called branches (just like for a complex logarithm). A whole multi-function is thence a union of all the possible branches. From a point of applications, the most interesting are two real valued branches, denoted by W_0 (principal branch, sometimes referred as W_p) and W_{-1} (negative branch, sometimes referred as W_m). The first one takes real values in the interval $[-e^{-1}, \infty)$, (it is defined for all complex numbers; $-e^{-1}$ is its branch point and the halfline $(-\infty, -e^{-1})$ is its branch cut) and the second in $[-e^{-1}, 0)$ (the branch point for all non-principal branches of W-Lambert function is always 0 and theirs branch cuts are halflines $(-\infty, 0)$) (compare with Figure 1.1).

It is implemented in all the most popular mathematical software packages (e.g. ProductLog in Wolfram Mathematica[19], lambertw in Matlab and Octave or LambertW in Maple).

Fact 1.2.14. *For $|x| < \frac{1}{e}$, the foregoing formula is satisfied (from [26]):*

$$W_0(x) = \sum_{i=1}^{\infty} (-1)^{i-1} \frac{i^{i-2}}{(i-1)!} x^i \tag{1.8}$$

[19] An inversion of e^x function is a logarithm. Moreover W-Lambert function is an inversion of product of x and e^x. Therefore this is the genesis of the function name in Wolfram Mathematica.

Fact 1.2.15. *The beneath formula give a useful approximation for the positive branch of W-Lambert function:*

$$W_0(x) = \ln(x) - \ln(\ln(x)) + \mathrm{O}\left(\frac{\ln(\ln(x))}{\ln(x)}\right) , \tag{1.9}$$

when $x \to \infty$.

Fact 1.2.16. *When $x < 0$, then the beneath formula give a useful approximation for negative W-Lambert function:*

$$W_{-1}(x) = \ln(-x) - \ln(-\ln(-x)) + \mathrm{O}\left(\frac{\ln(-\ln(-x))}{\ln(-x)}\right) , \tag{1.10}$$

as $x \to 0^-$

The above facts can be found in [26] or [1]. One can also check them for more properties and details.

1.2.13 Bernoulli numbers and polynomials

Definition of Bernoulli numbers

In 1713, Jacob Bernoulli defined a sequence of numbers, which commonly occur in combinatorics. Let us provide a definition of these numbers by its exponential generating function:

$$\frac{t}{e^t - 1} = \sum_{n=0}^{\infty} B_n \frac{t^n}{n!} .^{20}$$

By calculation of derivatives of both sides, we can deduce the values of $(B_n)_n$ sequence. Astonishingly, Bernoulli numbers vanish for odd indices except for B_1. Moreover, all values of this sequence are rational. In Table 1.3 we present non-zero values of the Bernoulli sequence for the smallest indices. One can realize an elegant property of Bernoulli numbers, namely $B_{4n+2} > 0$ and $B_{4n+4} < 0$ for $n \in \mathbb{N}_0$.

n	0	1	2	4	6	8	10
B_n	1	$-\frac{1}{2}$	$\frac{1}{6}$	$-\frac{1}{30}$	$\frac{1}{42}$	$-\frac{1}{30}$	$\frac{5}{66}$

Table 1.3: First non-zero values of the Bernoulli sequence $(B_n)_n$

[20] There exist also another, slightly different sequences with the same name. They differs in signs of some elements.
The left hand side of the equation is often referred to as Bernoulli function.

Definition of Bernoulli polynomials

Bernoulli numbers are closely related to Bernoulli polynomials, which can be defined by the following exponential generating function:

$$\frac{te^{xt}}{e^t-1}=\sum_{n=0}^{\infty}B_n(x)\frac{t^n}{n!}\ . \tag{1.11}$$

However, an explicit formula, given by the Bernoulli numbers is more wieldy in practise:

$$B_n(x)=\sum_{k=0}^{n}\binom{n}{k}B_{n-k}x^k\ .$$

We may briefly see that:

$B_0(x)=1$,

$B_1(x)=x-\frac{1}{2}$,

$B_2(x)=x^2-x+\frac{1}{6}$,

$B_3(x)=x^3-\frac{3}{2}x^2+\frac{1}{2}x$,

$B_4(x)=x^4-2x^3+x^2-\frac{1}{30}$.

Note that $B_n(0)=B_n$.

A useful bound on the values of Bernoulli polynomial in the interval $[0,1]$ is presented e.g. in [73]:

Fact 1.2.17.

$$|B_n(x)|\leqslant\frac{2n!}{(2\pi)^n}\zeta(n)\ .^{21} \tag{1.12}$$

1.2.14 Faulhaber's formula and Euler summation formula

A sum of powers of consecutive natural numbers were investigated since ancient times (e.g. by Pythagoras or Archimedes). However, in XVII-th century Johann Faulhaber proposed to give the result in a form of polynomials (see e.g. [71]). Formulas provided by Faulhaber were rectified by Jacob Bernoulli in his work from 1713 [12] (however, without a proof). Let $p\in\mathbb{N}$. A sum of p-powers of natural numbers, known as Faulhaber's formula is given as:

$$\sum_{k=1}^{n}k^p=\frac{n^{p+1}}{p+1}+\frac{n^p}{2}+\sum_{k=2}^{p}\frac{B_k}{k!}[p]_{k-1}n^{p-k+1}\ .^{22} \tag{1.13}$$

[21] for $n\neq 0$

[22] Note that if we change the definition of B_1 to $\frac{1}{2}$, then the right hand side is $\sum_{k=0}^{p}\frac{B_k}{k!}[p]_{k-1}n^{p-k+1}$.

The Bernoulli numbers for even indices combined with binomial coefficients increase in magnitude very rapidly and alternate in sign as well, so providing a good approximation of sums of relatively big powers p may be challenging.

A useful extension of Faulhaber's formula is so called Euler—Maclaurin summation formula (see e.g. [54, 4]).

Theorem 1.2.2 (Euler—Maclaurin formula)**.** *Let $a, b, s \in \mathbb{N}$, f be continuous, integrable, real (or complex) valued function on $[a, b]$. Then*

$$\sum_{i=a}^{b} f(i) = \int_a^b f(x)dx + \frac{f(b)+f(a)}{2} + \sum_{i=1}^{\lfloor \frac{s}{2} \rfloor} \frac{B_{2i}}{(2i)!} \left(f^{(2i-1)}(b) - f^{(2i-1)}(a) \right) + r_s \ , \tag{1.14}$$

where the remainder is given by:

$$r_s = (-1)^{s+1} \int_a^b f^{(s)}(x) \frac{B_s\left(\{x\}_f\right)}{s!} dx \ . \tag{1.15}$$

1.3 Probability Theory

1.3.1 General definitions and notations

In probability theory notation often varies from source to source, therefore we present in here a concise introduction to this field, together with utilized symbols.

A *universe* Ω denotes a set of all *elementary events* ω. A *probability of an event* A is denoted by $\Pr[A]$. We denote random variable X by uppercase letter, and its realizations either by the relevant lower-case letter x or $X(\omega)$, whenever ω is inherently necessary. When X has a *distribution* D with a vector of *parameters* Θ, then we denote such the relation as $X \sim D(\Theta)$. By $\mathbb{E}[X]$ and $\mathbb{V}\mathrm{ar}[X]$ we denote respectively an *expected value* and a *variance* of a random variable X. Its *cumulative distribution function* (abbreviated to CDF) we usually depict by F_X (i.e. $F_X(x) = \Pr[X \leqslant x]$)[23].

- If F_X is absolutely continuous with respect to Lebesgue measure on $\mathbb{R}$, then X has a *probability density function*(or shortly pdf[24]) and it is denoted by f_X (i.e. $f_X(x) = F'_X(x)$). In this case, a set $\{x : f_X(x) > 0\}$ is a *support* of X.

- If X is a discrete random variable, then an *atom* of X is such x that $p_X(x) := F_X(x) - \lim_{t \to x^-} F_X(t) > 0$.[25] In this case, a set of all atoms of X

[23] We assume that CDF is càdlàg, i.e. right continuous with left limits

[24] The term "pdf" is also a common abbreviation for "partial differential equation", however in this thesis it has only one meaning.

[25] When X is a discrete random variable and A is a set of all atoms of X, then $\sum_{x \in A} p_X(x) = 1$.

is its *support*, $p_X(x)$ denotes a *mass* of atom x and p_X is a *probability mass function* of X (abbreviated to pmf). In some cases a domain of pmf is given a priori. Then each point x from this domain, for which $p_X(x) = 0$ is called a *null atom*.

In both cases we denote the support of X as $\mathrm{Range}(X)$. When a random variable is known from the context, then we often omit the subscripts and simply write $F(x)$, $f(x)$ and $p(x)$.

Fact 1.3.1. *If* $\Pr[X \geqslant 0] = 1$, *then*

- $\mathbb{E}[X] = \int_0^\infty \Pr[X \geqslant x]\, dx$, *if* X *has probability density function,*
- $\mathbb{E}[X] = \sum_{k=1}^{\infty} \Pr[X \geqslant k]$, *if* X *has probability mass function.*

By $X|Y$ we denote a *conditional random variable*, which satisfies

$$\Pr[X \in A | Y \in B] \Pr[Y \in B] = \Pr[X \in A, Y \in B]$$

for every measurable A and B in ranges of X and Y respectively.

If a sequence (X_i) of random variables is *stochastically independent* and each X_i has the same distribution, then we write shortly that (X_i) are i.i.d. (independent and identically distributed).

A *simplex of probability distributions* on $[L]$ is defined as

$$\mathrm{Sim}_L := \left\{ \bar{p} \in [0,1]^L : \sum_{i=1}^{L} p_i = 1 \right\} .^{26}$$

A boundary of Sim_L consists of all the distributions on $[L]$, which have at least one null atom.

An infinite sequence $X_1, X_2, \ldots$ of real-valued random variables is said to *converge in distribution* (or *weakly*) if there exists a CDF F such that

$$\lim_{n\to\infty} F_{X_n}(x) = F(x) ,$$

for every $x \in \mathbb{R}$ such that the function F is continuous at x. We denote this fact by $F_{X_n} \xrightarrow[n\to\infty]{d} F$. When $(X_n)_n$ converges in distribution to X, then for every bounded and continuous function f, the following is satisfied:

$$\int_{\mathbb{R}} f(x) dF_{X_n} \overset{n\to\infty}{\longrightarrow} \int_{\mathbb{R}} f(x) dF_X .^{27}$$

[26] Let us point out that this notation is very similar to this of discrete simplex (see Section 1.2.4).

[27] See e.g. [14].

1.3.2 Order statistics

When $X_1, \ldots, X_m$ are independent random variables, then we can order them from the smallest, to the biggest. Then *order statistic* $\mathrm{Ord}_{i:m}(X_1, \ldots, X_m)$, for $i \in [m]$, describes the i-th smallest variable amongst them.

Fact 1.3.2. *If $X_1, \ldots, X_m$ are independent random variables and F is a common CDF of each X_i, then the CDF of $\mathrm{Ord}_{i:m}(X_1, \ldots, X_m)$ variable is given by the formula:*

$$\sum_{j=i}^{m} \binom{m}{j} (F(x))^j (1 - F(x))^{m-j} \ .$$

Note that in particular $\mathrm{Ord}_{1:m}(X_1, \ldots, X_m) = \min(X_1, \ldots, X_m)$ and symmetrically $\mathrm{Ord}_{m:m}(X_1, \ldots, X_m) = \max(X_1, \ldots, X_m)$, so one can use the above fact to obtain:

Fact 1.3.3. *If $X_1, \ldots, X_m$ are independent random variables and F is a common CDF of each X_i, then the CDF of $\max(X_1, \ldots, X_m)$ variable is F^m and the CDF of $\min(X_1, \ldots, X_m)$ is $1 - (1 - F)^m$.*

1.3.3 Stochastic processes with discrete time

A stochastic process X with discrete time is a sequence of random variables X_n, where n is element of either $[n]$ (for some $n \in \mathbb{N}$), $\mathbb{N}$ or $\mathbb{N}_0$. If X is a process with discrete time and has the property

$$(\forall\, n)\ \Pr\left[X_n \in A_n | \bigwedge_{i=1}^{n-1} (X_i \in A_i)\right] = \Pr[X_n \in A_n | X_{n-1} \in A_{n-1}] \ ,^{28}$$

whenever the above probabilities are well defined, then X is called a *Markov chain*. For more precise definition and properties, see e.g. [87].

1.3.4 Probabilistic Counters

The notion of probabilistic counter is ambiguous, so for reader's convenience, we provide a general definition of probabilistic counter, in the following way:

Definition 1.3.1. *We call a stochastic process M a probabilistic counter if*

$$M_{n+1} = f(M_n, X(M_n))$$

where M_n is the value of the counter after n incrementation requests, $X(M_n)$ is a random variable, possibly dependent on M_n and f is an arbitrary, non-negative function.

[28] When we define the process on $\mathbb{N}_0$, then we should also consider X_0 in conditioning.

Note that, using this definition, any probabilistic counter can be described with a tuple $\{M_0, (f(\cdot,\cdot), X(.))\}$. Moreover, we sometimes consider a case, when probabilistic counter is interpreted as an mechanism, which receives some queries. Such the query may either provoke next incrementation request (denoted as '1') or induce a fake request (denoted as '0'), which does not change the state of the probabilistic counter.

In Figure 1.2 one can see a graphical representation of the probabilistic counter. 1 denotes the incrementation request and 0 denotes fake request. The dice depicts the randomness (namely $X(M_n)$) from Definition 1.3.1 and "Increment" means that the application of $f(M_n, X(M_n)) \neq M_n$, so the probabilistic counter changes its value.[29]

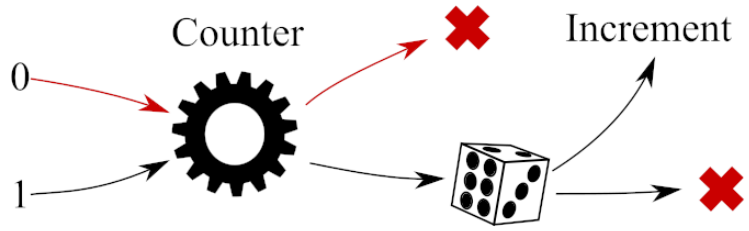

Figure 1.2: Graphical depiction of probabilistic counter.

Probabilistic counters are usually used in order to estimate some quantity G, like e.g. size of a database. Then there exists some estimator of the property G given by some function Ψ, i.e. $\Psi(M_n)$ is an estimate of G (which depends on n). Then it is important to measure an *accuracy* of the estimator Ψ. We define it in a standard way as $\frac{\sqrt{\mathbb{V}\mathrm{ar}[\psi(M_n)]}}{n}$.

1.3.5 Variants of probability distributions

Notions of some distributions are commonly ambiguous. Often distributions, which vary slightly, have the same name. In order to distinguish them in a convenient way, we introduce some terms and symbols, which indicates an actual variant.

First of all, let us define *restricted* (or *truncated*) version of variables. Namely, when $L \in \mathbb{N}$ and $X \sim D(\Theta)$, then we say that Y is a restricted version of random variable X if $Y = \max(X, L)$, i.e. we restrict X to the interval $(-\infty, L]$.[30] Then we denote that $Y \sim D(\Theta, L)$.

We also define *conditionally truncated* version of variables. Namely, let $X \sim D(\Theta)$, with some support S and $A \subset S$. We say that Y is called a version of X

[29] Probabilistic counters are usually non-decreasing with respect to the number of incrementation request n, hence we signed this step as "Increment". However, sometimes they are also allowed to reset the value, so the nomenclature remained unchanged, although we have generalized the definition of probabilistic counter to Definition 1.3.1.

[30] In fact, we usually consider X with support $\mathbb{N}$ or $\mathbb{N}_0$, so we restrict it to $[L]$ or $[0 : L]$ respectively.

conditionally truncated to A, if $Y = (X|X \in A)$. It implies, for instance, that for $B \subset A$, $\Pr[Y \in B] = \dfrac{\Pr[X \in B]}{\Pr[X \in A]}$. Then we denote $Y \sim (D(\Theta)|A)$.[31]

Let us also define *shifted* versions of discrete distributions. Some of distributions have the same name for the same concept, but "shifted" by one in either of two sides of real line. Let $X \in D(\Theta)$ be discrete random variable. Then $Y = X + 1$ is called R-shifted version of X and $Z = X - 1$ is L-shifted version of X. We denote these facts as $Y \sim D_{\rightarrow}(\Theta)$ and $Z \sim D_{\leftarrow}(\Theta)$ respectively.[32]

We are going to use the same names of variants for the distributions as well. For instance, we will consider G with R-shifted version of geometric distribution, conditionally truncated to $[n]$.

1.3.6 Uniform Distributions

We denote a (discrete) uniform distribution on the set $[n]$ by $\mathrm{Uni}\,(n)$. If $U \sim \mathrm{Uni}\,(n)$, then $\Pr[U = k] = \frac{1}{n}$ for $k \in [n]$.

We say that random variable U has a (discrete) uniform distribution on a finite set S of size n, if $\Pr[U = s] = \frac{1}{n}$ for any $s \in S$. We denote it shortly by $U \sim \mathrm{Uni}\,(S)$. Especially, when we are drawing a random bit uniformly, it states for a choice from a set $S = \{0, 1\}$. Note that $\mathbb{E}[U]$ is a mean of the values from the set S. Especially, when $S = [n]$, we get $\mathbb{E}[U] = \frac{n+1}{2}$.

Let $a < b$. By $\mathrm{Uni}\,(a, b)$ we denote a (real-valued) uniform distribution on the interval (a, b). If $U \sim \mathrm{Uni}\,(a, b)$, then pdf of U is $f_U(x) = \frac{1}{b-a}\mathbf{1}_{(a,b)}\,(x)$. Then simply $\mathbb{E}[U] = \frac{a+b}{2}$. If A is Borel-measurable subset of $\mathbb{R}$ with finite, non-zero Lebesgue measure, then we denote $U \sim \mathrm{Uni}\,(A)$ with pdf $f_U(x) = \frac{1}{\lambda(A)}\mathbf{1}_A\,(x)$.

Theorem 1.3.1 (Inverse transform sampling (Smirnov method))**.**

1. *Let F_X be absolutely continuous CDF of X. Then $U = F_X(X)$ has $\mathrm{Uni}\,((0, 1))$ distribution.*
   If $U \sim \mathrm{Uni}\,((0, 1))$, then $F_X^{-1}(U)$ has the same distribution as X.

2. *Let X has discrete distribution. If $U \sim \mathrm{Uni}\,((0, 1))$, then $\min\{x : U \leqslant F_X(x)\}$ has the same distribution as X.*

1.3.7 Bernoulli, Binomial and Normal distributions

A random variable B has Bernoulli distribution $\mathrm{Ber}\,(p)$ if $\Pr[B = 1] = p = 1 - \Pr[B = 0]$. Then $\mathbb{E}[B] = p$ and $\mathbb{V}\mathrm{ar}[B] = p(1 - p)$.

[31] Note that terms "conditionally truncated" and "truncated" may be a little bit confusing. Nevertheless, we decided to only add an adjective "conditionally" in order to distinguish this case from the one given by rectification of the distribution via max operation and to emphasize that the resulting distribution is obtained by conditioning the support of the initial one in contrast to the second variant.

[32] To abbreviate the notation, we sometimes evoke them as R-versions and L-versions of random variables.

A random variable S has Binomial distribution $\mathrm{Bin}\,(n,p)$ if it can be given as a sum of n i.i.d. random variables with $\mathrm{Ber}\,(p)$ distribution. Naturally, $\mathbb{E}[S]=np$ and $\mathbb{V}\mathrm{ar}[S]=np(1-p)$.

A random variable N has Normal distribution $\mathcal{N}\,(\mu,\sigma)$ if it has pdf

$$f(x)=\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)\ .$$

Then $\mathbb{E}[N]=\mu$ and $\mathbb{V}\mathrm{ar}[N]=\sigma^2$. A distribution $\mathcal{N}\,(0,1)$ is called standard Normal distribution. Cumulative distribution function of $\mathcal{N}\,(0,1)$ is a special function denoted by Φ.[33] In Chapter 2 will utilize the fact, that $\Phi^{-1}(0.99)\approx 2.325$.

Theorem 1.3.2 (de Moivre—Laplace)**.** *Let $(B_i)_{i=1}^{\infty}$ be a sequence of i.i.d. random variables with* $\mathrm{Ber}\,(p)$ *distribution and $N\sim\mathcal{N}\,(0,1)$. Then*

$$\frac{\sum\limits_{i=1}^{n}B_i-np}{\sqrt{np(1-p)}}\xrightarrow[n\to\infty]{d}N\ .$$

Let us mention that Theorem 1.3.2 is the simplest version of Central Limit Theorem.

1.3.8 Exponential and Laplace distributions

A random variable E has an exponential distribution with parameter $\lambda>0$ (denoted by $E\sim\mathrm{Exp}\,(\lambda)$) if its support is $[0,\infty)$ and $\Pr[E>x]=e^{-\lambda x}$ for each $x\geqslant 0$. A pdf of this distribution is $f_E(x)=1_{[0,\infty)}\,(x)\,\lambda e^{-\lambda x}$. If $E\sim\mathrm{Exp}\,(\lambda)$, then $\mathbb{E}[E]=\frac{1}{\lambda}$, which is called an intensity of exponential distribution.[34]

A random variable X has a Laplace distribution with parameter $\lambda>0$ (denoted by $X\sim\mathcal{L}(\lambda)$), if its probability density function is

$$f(x)=\frac{1}{2\lambda}\exp\left(-\frac{|x|}{\lambda}\right)\ .$$

Then $\mathbb{E}[X]=0$ and $\mathbb{V}\mathrm{ar}[X]=2\lambda^{-2}$.

1.3.9 Beta distribution

A random variable B has Beta distribution with parameters $a,b>0$ ($B\sim\mathrm{B}\,(a,b)$) if its support is $[0,1]$ and its density is given by the function $f_B(x)=\frac{x^{a-1}(1-x)^{b-1}}{\mathrm{B}(a,b)}$. If $B\sim\mathrm{B}\,(a,b)$ then $\mathbb{E}[B]=\frac{a}{a+b}$.

[33] Φ is closely related to, so called, error functions. For definition and properties, see e.g. [1].

[34] Note that in the literature, λ parameter is often the intensity of the distribution.

1.3.10 Geometric-Like Distributions

Let us fix a number $p \in (0,1)$ and let $q = 1-p$. A random variable G with values in $\mathbb{N}$ has a geometric distribution with parameter p (denoted by $G \sim \mathrm{Geo}(p)$) if $\Pr[G = k] = pq^k$ for $k \in \mathbb{N}_0$. It represents the first moment of success in a process, where in each step, the probability of success is p. $\mathbb{E}[G] = \frac{q}{p}$

A random variable X has a restricted geometric distribution with parameters $p \in (0,1)$ and $L \in \mathbb{N}$ (indicated by $X \sim \mathrm{Geo}(p, L)$) if there exists a random variable $G \sim \mathrm{Geo}(p)$ such that $X = \min(G, L)$.

Fact 1.3.4. *Notice that if $X \sim \mathrm{Geo}(p, L)$, then*

$$\Pr[X = k] = \begin{cases} pq^k & \text{if } k \in [0 : L-1] \\ q^L & \text{if } k = L \ . \end{cases}$$

A random variable M has maximal geometric distribution $\mathrm{MGeo}(n, p)$ (usually abbreviated to "MaxGeo") if there exist i.i.d. random variables $X_1, \ldots, X_n$, with the distribution $\mathrm{Geo}(p)$, such that

$$M = \max(X_1, \ldots, X_n) \ .^{35}$$

Fact 1.3.5. *For $k \in \mathbb{N}$, $\Pr[M \leqslant k] = \left(1 - p^k\right)^n$.*

Random variables with $\mathrm{MGeo}(n, p)$ distribution were studied in several papers (see e.g. [99, 22]).

Fact 1.3.6. *If $M \sim \mathrm{MGeo}(n, p)$, then $\mathbb{E}[M] = \frac{1}{2} + \frac{\mathrm{H_n}}{-\ln(1-p)} + P(n) + \mathrm{O}\left(\frac{1}{n}\right)$, where $P(n)$ is a periodic function with small amplitude.*

From equations (1.7) and (1.5), we attain $\mathbb{E}[M] \sim \frac{1}{2} + \frac{\ln n + \gamma}{p}$, as $p \to 0^+$ and $n \to \infty$.

Fact 1.3.7. *If $M \sim \mathrm{MGeo}(n, p)$ and $C > 0$ is some constant, then*

$$\Pr\left[M \geqslant C\frac{\ln n}{\ln Q}\right] \leqslant \frac{1}{n^{C-1}} \ ,$$

where $Q = \frac{1}{1-p} = q^{-1}$.

Proof. If $G \sim \mathrm{Geo}(p)$ and $k \in \mathbb{N}$, then $\Pr[G \geqslant k] = q^k$. Therefore $\Pr[M \geqslant k] \leqslant nq^k$,[36] hence

$$\Pr\left[M \geqslant C\frac{\ln n}{\ln Q}\right] \leqslant n \exp\left(C\frac{\ln n}{\ln Q} \ln q\right) = \frac{1}{n^{C-1}} \ .$$

□

[35] In Chapter 4 we will use MaxGeo name for a variable with a distribution $\mathrm{MGeo}_{\rightarrow}(n, p)$. However it will be clearly pointed out.

[36] By inclusion—exclusion principle.

Definition 1.3.2. *A random variable $W_{n,p}$ has a distribution* $\mathrm{WMGeo}(n,p)$ *if there are independent random variables $G_1,\ldots,G_n$ with distribution* $\mathrm{Geo}(p)$ *such that*

$$W_{n,p} := \mathrm{card}\left(\{k : G_k = \max(G_1,\ldots,G_n)\}\right) \ .$$

Using the Mellin transform or the Rice method one can derive the following approximation (see e.g. [68, 21] for details):

Theorem 1.3.3. *Let $a,n \in \mathbb{N}$, $p \in \left(0,\frac{1}{2}\right)$ and $W_{n,p} \sim \mathrm{WMGeo}(n,p)$. Then the distribution of $W_{n,p}$ can be estimated by:*

$$\Pr[W_{n,p} = a] = -\frac{1}{\ln(1-p)}\frac{p^a}{a} + r_p \ ,$$

where $|r_p| < -\frac{(a+1)^2}{12a}p^a\ln(1-p)$.

Fact 1.3.7 may be used to find such a number k that $\Pr[W_{n,p} > k]$ is very small and Theorem 1.3.3 will be utilized in Section 2.7.2 for finding the probability of success in LEAs based on geometric distributions.

Remark It worth to notice that the right side of formula from Theorem 1.3.3 makes an impression of being independent of n. In fact the number $\Pr[W_{n,p} = a]$ depends on n, but the influence of n is negligibly small for all $p < \frac{1}{2}$ (see [21]) and is hidden in the expression r_p. Also notice that from Theorem 1.3.3 it follows that $\Pr[W_{n,p} = 1] = 1 - \frac{p}{2} + \mathrm{O}\left(p^2\right)$ as p tends to 0 (compare with a notice about the probability of collision from Section 2.7.2).

We introduce auxiliary notations related to the distribution $\mathrm{WMGeo}(n,p)$. Namely, let us define $W_{k,n,p} := \Pr[W_{n,p} = k]$ and $W_{\geqslant 2,n,p} := \Pr[W_{n,p} \geqslant 2]$, where $W_{n,p} \sim \mathrm{WMGeo}(n,p)$. Obviously, then we may determine $W_{1,n,p} = 1 - W_{\geqslant 2,n,p}$.

Beneath we present one of the results from [21]:

Theorem 1.3.4. *Let $a,n \in \mathbb{N}$, $p \in \left(0,\frac{1}{2}\right)$ and $W_{n,p} \sim \mathrm{WMGeo}(n,p)$. Then the distribution of $W_{n,p}$ can be bounded by:*

$$\Pr[W_{n,p} \geqslant a] < -\frac{p^a}{a(1-2p)\ln(1-p)} - \frac{(a+1)^2 p^a \ln(1-p)}{12a(1-2p)} \ .$$

We say that X has R-shifted geometric distribution, conditionally truncated to $[n]$, if its probability mass functions is given by the formula $\Pr[X = k] = \frac{pq^{k-1}}{1-q^n}$, for $k \in [n]$, $p \in (0,1)$ and $q = 1-p$. We denote it as $X \sim (\mathrm{Geo}_{\rightarrow}(p)|[n])$. Note that this distribution is different from $\mathrm{Geo}(p,n)$.

1.3.11 Zipf Distribution

We say that X has Zipf $Z(n,m)$ distribution, if $\Pr[X = k] = \frac{1}{k^m H_n{}^{(m)}}$, for $k \in [n]$, where $m \geqslant 1$ is fixed. Then $F(k) = \frac{H_k{}^{(m)}}{H_n{}^{(m)}}$, for $k \in [n]$. Naturally, $\mathbb{E}[X] = \frac{H_n{}^{(m-1)}}{H_n{}^{(m)}}$.

1.3.12 Poisson Distribution

Let $\lambda > 0$. Then X has Poisson Pois(λ) distribution, if $\Pr[X = k] = \frac{\lambda^k}{k!}e^{-\lambda}$, for $k \in \mathbb{N}_0$. Moreover $\mathbb{E}[X] = \lambda$.

We say that X has R-shifted Poisson distribution, conditionally truncated to $[n]$, if $\Pr[X = k] = \frac{\lambda^{k-1}}{a_n(k-1)!}e^{-lm}$, for $k \in [n]$, where $a_n = \sum_{i=0}^{n-1} \frac{\lambda^i}{i!}e^{-\lambda}$ is a normalizing constant. We denote it as $X \sim (\mathrm{Pois}_{\rightarrow}(\lambda)|[n])$.

1.3.13 Distance between discrete probability distributions

Denote $\Pr[X \in A]$ by $P_X(A)$. Let X and Y be defined on the common probability space $\mathcal{X} = (\Omega, \mathcal{F}, \mathbb{P})$.

Total Variation distance

We define *total variation distance* between P_X and P_Y as $\sup_{A\in\mathcal{F}} |P_X(A) - P_Y(A)|$, denoted by $\|P_X, P_Y\|_{TV}$. Alternatively, $\|P_X, P_Y\|_{TV}$ can be written by the norm in ℓ_1 space as $\frac{1}{2}\|P_X - P_Y\|_1$.

Kullback—Leibler divergence

Let X and Y have a common support. *Kullback—Leibler divergence* (or *relative entropy*) from P_Y to P_X is defined as

$$D_{KL}(P_X\|P_Y) = \sum_{i\in\mathrm{Range}(X)} P_X(\{i\}) \lg\left(\frac{P_X(\{i\})}{P_Y(\{i\})}\right) .$$

It worth to mention that Kullback—Leibler divergence is not symmetric, so usually $D_{KL}(P_X\|P_Y) \neq D_{KL}(P_Y\|P_X)$.

Fact 1.3.8. *Relation between total variation distance and relative entropy is given by the inequality:* $\|P_X, P_Y\|_{TV} \leqslant \sqrt{\frac{1}{2}D_{KL}(P_X\|P_Y)}$.

1.4 Useful standalone theorems

1.4.1 Banach fixed point theorem

Theorem 1.4.1 (Banach fixed point theorem)**.** *Let $(\mathcal{X}, d)$ be a non-empty complete metric space with mapping $T : \mathcal{X} \to \mathcal{X}$ with contraction constant*

$$\Lambda := \sup\{\lambda \in \mathbb{R} : (\forall\, x, y \in \mathcal{X})\ d(T(x), T(y)) \leqslant \lambda d(x, y)\} .$$

Then T admits a unique fixed-point $x^* \in \mathcal{X}$. Furthermore, $x^* = \lim_{n\to\infty} x_n$ where x_0 is an arbitrary element of $\mathcal{X}$ and $x_n = T(x_{n-1})$ for $n \geqslant 1$. $(x_n)_n$ is called a Banach sequence. Then also

$$d(x^*, x_n) \leqslant \frac{\Lambda^n}{1-\Lambda} d(x_1, x_0) .$$

1.4.2 Weierstrass' Product Inequality

Theorem 1.4.2 (Extension of Weierstrass' Product Inequality)**.** *(see [69] for a proof) Let I be some finite set and $(\forall\, i \in I)\ a_i \in [0,1]$. Then*

$$1 - \sum_{i\in I} a_i \leqslant \prod_{i\in I}(1-a_i) \leqslant 1 - \sum_{i\in I} a_i + \sum_{i\in I}\sum_{j\in I\setminus\{i\}} \frac{1}{2} a_i a_j \ .$$

We will refer to Theorem 1.4.2 shortly via "WPI" (usually as an indicator over the $\leqslant$ symbol). We are going to provide a more general result, which was originally presented in [69] via elementary symmetric polynomials. Nevertheless we are going to present it together with a short proof in an other form utilizing multi-indices. A multi-index of order n is a vector $\kappa \in \mathbb{N}^n$. We define $|\kappa| = \sum_{i=1}^{n} \kappa_i$ and $\max\{\kappa_i : i \in [n]\}$ to be respectively length and depth of κ. If $x \in \mathbb{C}^n$ and $\kappa \in \mathbb{N}^n$, then we write shortly x_κ instead of $\prod_{i=1}^{n} x_i^{\kappa_i}$.
If κ, ι are multi-indices, then we write $\kappa\|\iota$ for theirs concatenation.[37] A set of all multi-indices of order n, length L and depth at most d is denoted as $M(n,L,d)$. We also define $\bowtie_r$ to be $\leqslant$, when r is even number and $\geqslant$, when r is odd.

Theorem 1.4.3 (Generalized Weierstrass' Product Inequality)**.** *For $r \leqslant n$ and a sequence $(x_i) \in [0,1]^n$, the following inequalities are satisfied:*

$$\prod_{i=1}^{n}(1-x_i) \bowtie_r \sum_{L=0}^{r} \sum_{\kappa\in M(n,L,1)} (-1)^L x_\kappa \ . \tag{1.16}$$

Proof. We use the induction with respect to n. A standard combinatorial argument shows that when $n = r$, then the equality holds.
Assume, that the formula (1.16) is true for some $n \in \mathbb{N}$. We will show, that the similar will be true for $n+1$. Indeed,

$$\begin{aligned}
\prod_{i=1}^{n+1}(1-x_i) &\overset{x_{n+1}\leqslant 1}{\bowtie_r} (1-x_{n+1}) \sum_{L=0}^{r} \sum_{\kappa\in M(n,L,1)} (-1)^L x_\kappa \\
&= 1 + \sum_{L=1}^{r} \sum_{\kappa\in M(n,L,1)} (-1)^L x_\kappa - x_{n+1} \sum_{L=0}^{r} \sum_{\kappa\in M(n,L,1)} (-1)^L x_\kappa \\
&= 1 + \sum_{L=1}^{r} \sum_{\kappa\in M(n,L,1)} (-1)^L x_{\kappa\|(0)} + \sum_{L=0}^{r-1} \sum_{\kappa\in M(n,L,1)} (-1)^{L+1} x_{\kappa\|(1)} \\
&\quad - x_{n+1} \sum_{\kappa\in M(n,r,1)} (-1)^r x_\kappa = \dots
\end{aligned}$$

[37] Just like in the definition from Section 1.2.1.

$$\ldots = 1 + \sum_{L=1}^{r} \sum_{\kappa \in M(n+1,L,1)} (-1)^L x_\kappa - x_{n+1} \sum_{\kappa \in M(n,r,1)} (-1)^r x_\kappa$$
$$\overset{\substack{x_{n+1} \geqslant 0 \\ \bowtie_r}}{} \sum_{L=0}^{r} \sum_{\kappa \in M(n+1,L,1)} (-1)^L x_\kappa \ .$$

□

In fact, the formula (1.16) and the above proof are more explicit, precise and concise than relevant equivalents from [69].[38]

1.4.3 Cauchy—Schwarz inequality

Theorem 1.4.4 (Cauchy—Schwarz inequality)**.** *Let x, y be vectors from a unitary space with norm $\|.\|_2$. Then*

$$|\langle x, y \rangle| \leqslant \|x\|_2 \cdot \|y\|_2 \ .$$

Especially, if $x, y \in \ell_2(\Omega)$, then

$$\left| \sum_{i \in \Omega} x_i y_i \right| \leqslant \sqrt{\sum_{i \in \Omega} x_i^2} \cdot \sqrt{\sum_{i \in \Omega} y_i^2} \ .$$

1.4.4 Weierstrass Extreme Value Theorem

Theorem 1.4.5 (Weierstrass Extreme Value Theorem)**.** *If $C \in \mathbb{R}^d$ is bounded and closed set and $f : C \to \mathbb{R}$ is a continuous function, then there exist $a, b \in C$ such that*

$$(\forall\, x \in C)\ f(a) \leqslant f(x) \leqslant f(b).$$

1.4.5 Stević Theorem

Theorem 1.4.6 (From [96])**.** *Let $f\colon (0, \alpha) \to (0, \alpha)$, where $\alpha > 0$, be a continuous function such that $(\forall\, x \in (0, \alpha))\ 0 < f(x) < x$ and $f(x) = x - ax^k + bx^{2k-1} + \mathrm{o}\left(x^{2k-1}\right)$, when $x \to 0^+$, where $k > 1$, $a, b > 0$. Let $x_0 \in (0, \alpha)$ and $x_{n+1} = f(x_n)$ for $n \in \mathbb{N}_0$. Then*

$$x_n = (a(k-1)n)^{-\frac{1}{k-1}} - \frac{(ka^2 - 2b)\ln n}{2(a(k-1))^2 (a(k-1))^{\frac{1}{k-1}} n^{\frac{k}{k-1}}} + \mathrm{o}\left(\frac{\ln n}{n^{\frac{k}{k-1}}}\right) \ .$$

1.4.6 Bernoulli Inequality

Theorem 1.4.7. *Let $x \geqslant -1$ and $k \geqslant 1$. Then*

$$(1 + x)^k \geqslant 1 + kx \ .$$

[38] In our proof there is other base case of the induction.

1.4.7 Fatou's Lemma

Lemma 1 (Fatou's Lemma)**.** *Let $f_n : X \to [0, \infty)$ be a sequence of Lebesgue-measurable functions on X. Then*

$$\int_X \liminf_{n\to\infty} f_n(x)dx \leqslant \liminf_{n\to\infty} \int_X f_n(x)dx \ .$$

1.4.8 Tonelli's Theorem

Theorem 1.4.8 (Discrete Tonelli's Theorem)**.** *Let I, J be (at most) countable sets. and $a(i, j) \geqslant 0$ for all $(i, j) \in I \times J$. Then*

$$\sum_{i\in I}\sum_{j\in J} a(i,j) = \sum_{j\in J}\sum_{i\in I} a(i,j) \ .$$

There exists also a more general version of Tonelli's Theorem for functions defined on product of σ-finite measure spaces. This result is closely related to Fubini's Theorem.

1.5 Other

1.5.1 A brief introduction to graph theory

A simple graph G is a pair (V, E), where V is a set, which elements are called *vertices* and E is a relation on V, which is antireflexive $((\forall\ v \in V)\ \neg(v, v) \in E)$ and symmetric $((\forall\ v, w \in V)\ (v, w) \in E \Rightarrow (w, v) \in E)$. An element (v, w) of E is called an *edge*. $\mathrm{card}\,(V)$ is called an *order of the graph*. A graph, in which $E = V \times V$ is called a *clique* or a *complete graph*. A set $N_v := \{w \in V : (v, w) \in E\}$ is called a *neighbourhood* of v. $\mathrm{card}\,(N_v)$ is called a *degree of vertex* v. A sequence of vertices $(v_1, v_2, \ldots, v_k)$ is called a *path* if $(\forall\ i \in [k-1])\ (v_i, v_{i+1}) \in E$. We say that the graph is *connected* if for all $v, w \in V$, there exists a path between v and w. Otherwise we call it *non-connected*. If all vertices has degree 2 and the graph is connected, then the graph is called a *ring*. $H = (V', E')$ is called an *induced subgraph* of $G = (V, E)$, if $E\big|_{V'\times V'} = E'$.

1.5.2 Networks

A network N is a communication structure with a topology given by a graph G_N. The vertices $V(G_N)$ are called nodes and the edges $E(G_N)$ are called ports. Each node may be interpreted as a device or a user and ports between them may be interpreted as the possible channels of communication (i.e. one device may send some data to the other one). It may be either assumed that a node communicates his message via all the neighbouring ports at the same moment or by one of the neighbouring ports. In the second approach, usually the sequence of communication queries are planned by routing tables. We may

itemize different properties of the networks. Let us notice that some of the properties of the graph G_N may be not known a priori to the nodes. For example, nodes order of the network $\mathrm{card}\,(V(G_N))$ may not be known to all of the nodes at the very beginning of an execution of algorithm. However, it is possible that some of the nodes will attain $\mathrm{card}\,(V(G_N))$ (or other properties) during the communication. We will delve into this distinction in Section 2.1. We usually denote the order of G_N by n. Another intrinsic property is a diameter of the network, which is a maximal number of hops that are needed to transmit any message. In other words it is the maximal distance in the topology induced by the graph G_N. We say that a network is static if the topology of the graph G_N either does not change or routing table of the network (whenever it exists) is established only once and remain unchanged.[39] Otherwise, i.e. when the topology of the network change in time, we describe it as dynamic network. A specific type of dynamic network is Ad Hoc network, in which there is no structure given a priori and the topology of the network can evolve significantly over time. This arrangement is typical for e.g. mobile wireless networks.

[39] A routing table consists of lists of different neighbours of the particular node, which defines the direction of a transfer of a message to any device in the network. For different destination points, the corresponding list can be different. If the port of the first choice on the list is unavailable for some reason (e.g. a technical issue), the next port on the list is used instead, etc.

Chapter 2

Leader Election Algorithms

2.1 Introduction and motivation

Commence our considerations with an arrangement, where a *group* of some *devices*[1] like e.g. computers, stations of radio-network, robots, drones, vehicles, register processes or household appliances, cannot establish a consensus.

Example 1. *As an introduction to a framework of a problem, let us consider two autonomous vehicles, which ride side by side and want to share the same road lane at the same time, what inevitably will cause a road accident. For instance, such the issue can be solved in the following trivial way: a vehicle A decides which of the vehicles should use the disputed lane, further A communicate the decision to a vehicle B, which agrees with the statement and proceeds accordingly to proposed recommendations.*

The above solution can be abstracted to a conception of Leader Election Algorithm (hereinafter referred to as LEA). It is a distributed process,[2] which executed by a group of conflicted devices, should provide a **unique** winner, who will be obliged to lead the group. Customarily such the node is called a *leader* or a *coordinator*.[3] When an execution of LEA correctly chooses a unique leader, then we call such the realization *successful*. Otherwise it is interpreted as a *failure*. Leader Election problem may be easily confused with a leadership election, in which the leader of some party should be chosen. We have to distinguish these two phrases, nevertheless LEA somehow encompass a case of leadership election, which can be conducted in the party to appoint leadership via appropriate execution of Leader Election Algorithm.

[1] Devices are often referred to as *processes* or *stations*, however we also use other names like *competitors*, *contestants*, *agents*, *appliances* or simply *nodes* (of the network).

[2] Distributed processing is an arrangement, in which algorithms can be executed by more than a single individual (e.g. appliances or register processes) in order to increase capabilities of a system of individuals.

[3] The leader is sometimes interpreted as a loser (like in [89]), since the coordinator usually lose more energy than the other agents.

A multitude of ideas for Leader Election Algorithms has brought a wide variety of scenarios, what motivated researchers to categorize them with respect to protocols' properties. However a nomenclature is not perfectly specified yet, so some of variants' names differ or even interfere.

In here we would like to provide a brief outline of various approaches, which we find the most crucial from the point of further considerations. Differing properties are mostly connected with a configuration, model of communication or a designation of the network.

From a configuration of the network point of view, there are two intrinsic distinctions. First of all, elections can be utilized both for unchanging configurations like *static networks*, where once connected devices remains in the network, as well as for the *dynamic* ones, like e.g. Ad Hoc networks, where appliances often change theirs connection statuses. When it is not stated straightforwardly, we assume that we deal with dynamic network, which is more general case. On the other hand, a structure of connections in the network influence the execution. The simplest possible setup is complete, where each of the processes can directly send information to all the others (so called *single-hop* arrangement). Otherwise, some of the communication has to be passed through some nodes in order to reach receiver, when direct bypasses are not available (*multi-hop* arrangement). These concepts are described e.g. in [10], nevertheless, we will also take a closer look on them in Section 2.4.1. Some of the solutions on Leader Election problem were designed for specific types of topology of the network (like e.g. trees, meshes, rings, toruses or hypercubes)[4]. However, in this dissertation we only consider so called *universal* Leader Election algorithms, which can be designated for any type of the topology. For completeness, let us mention about alternative model with a *population protocol*, which is quite similar to multi-hop arrangement, but in this case, every communication is performed via direct interactions, every realized by exchanging the internal states of two neighbouring stations (agents cannot communicate with more than one station at the time). Such the arrangement of LEAs was considered in plenty of papers (e.g. [2, 52, 97] to mention a few). Especially it worth to indicate [11], which provides optimal results with respect to the number of states and expected number of interactions. Nevertheless, we will not consider this population protocol routines.

It is straightforward that network's configuration has an inevitable impact on communication in the network and naturally should be taken into account. However, for a sake of clarity, in this dissertation, we do not pay much attention to this topic. We only mention here several from a broad spectrum of solutions. In order not to get rid of this problem too carelessly, we focus on very easy, low-level *beeping model*, introduced by Cornejo and Kuhn in [28] (we present the idea of the model in Section 2.4.1). This setup requires the communication to be divided into synchronized rounds.[5] The beeping model can be arrange with or without a *collision detection* (abbreviated to CD) option. In this dissertation

[4] See e.g. [93].

[5] Alternatively one may assume other models, where each device can send and receive messages at any time, optionally simultaneously (see e.g. [77]).

we consider a simpler version of the model, which do **not** utilize CD. A serious profit of our attitude is that the beeping model without CD can be simply adapted to more complicated variants of communication. We delve into this topic in Section 2.4.1. Let us point out, that there are many approaches to Leader Elections via beeping model both with CD (like e.g. in [107, 83]) and without CD (e.g. in [82]).

In this dissertation we always assume that all the communication is performed in a *single multi-access channel*, where every agent receives the same messages, if they are active and in the range of communication. However some of researchers consider more complicated, *multi-channel* model of communication as well (see e.g. the survey [61] for several approaches and challenges).

One of the most important aspects of Leader Election is to inform all the devices who the leader is. This task can be realized in universal LEAs via a natural framework by extrema propagation techniques (see e.g.[8, 9, 23]), initialized in a node, which becomes a leader, so we omit this part of LEAs in our considerations. Let us only remark, that under some circumstances this process can be coarsely simplified. For instance, for fully-connected networks the elected leader can directly send the information about his identity directly to all the other processes. We are going to raise the issue of extension to connected graphs in Section 2.4.1.

A second wide branch of properties of LEAs are based on theirs designation, which also can be partially dependent on the configuration of the network.

If a number of all competitors of LEA is unknown in advance, then we call it *anonymous* and when initially all the competitors know theirs total number n, then such LEA is called *non-anonymous*. There exists also a specification of anonymous algorithms, in which an upper bound N on the number of devices in the network is given, i.e. it is known that $n \in [N]$. We name such the number N as a *network's capacity*.[6] In order to distinguish this arrangement of the network, we name it quasi-anonymous. For completeness, if there are no size nor capacity of the network given a priori, then we call the leader election algorithm a *fully-anonymous* one. In order to simplify further descriptions, we designate similar names both for algorithms and networks, in which LEAs are performed. It worth to remark that, a fully-anonymous arrangement can be simply utilized as a quasi-anonymous, and in consequence, a non-anonymous one.

Note that the quasi-anonymous algorithms are strongly constrained, compared with non-anonymous ones. Therefore, either non-anonymous or fully-anonymous algorithms with unbounded time of termination, which mainly consider on asymptotic correctness and properties are mostly considered in the literature.

The request of quasi-anonymity is crucial for considerations in this dissertation, since it is inextricably linked with practical applications. Namely, we may use such the algorithms for any arbitrary subsets of the set of all nodes.

[6] Let us mention that "network's capacity" sometimes refers to the accumulated capacity of messages that can be transmitted via links of the network over time. However, in this dissertation we do not investigate such the details of the communication.

For example, we may consider only these nodes which are equipped with some specific sensors or which have sufficient energy resources or which observed some unusual phenomenon (e.g. a forming of eye of cyclone or a malfunction of some machine) are chosen to compete for the leader position.

Note, that if the network's configuration is dynamic, then it is highly probable that the devices do not know each other, so in particular then it can take some time to establish a total number of agents in such the setup. Hence, it is easier to consider that the network, like e.g. Ad hoc network, is anonymous. Remark that when some of the devices know the size of the network while the others are not aware of this size, then there is still a possibility to perform anonymous Leader Election.

We naturally assume that every contestant in the network wants to be the leader [7], so we should guarantee the election to be righteous. If all the competitors have equal chances to become the leader, then we say that LEA is *fair*. Otherwise we call such the algorithm *unfair*.

Imagine a situation when Leader Election is performed in some static network with few devices of the same type and the winner is the one with the biggest identity on the nameplate. If all the identifiers are distributed equally likely amongst all the devices, then such the election is fair. However, if we would like to repeat the elections, then the same device will win every time. Therefore we also specify an idea of *repetitive fairness* of LEA, which is fulfilled if the process is fair in every single, independent run. If the leader eventually resigns after some time after the election, then a new coordinator should be chosen and meanwhile a structure of the network may change (when the network is dynamic), like for instance, when energy of some devices have depleted. This argument affirms the assumption of repetitive fairness.

Researchers sometimes distinguish two kinds of fair algorithms: *oblivious* and "uniform" ones. It worth to mention, that there is also a type of algorithms, straightforwardly connected with those two, but irrelevant of fairness. They are called "non-uniform" ones. Descriptions of these three arrangements can be found e.g. in [82, 83]. Let us note that all these types assume that the Leader Election procedures are divided in synchronized *rounds* (*time steps*). An oblivious algorithm is the routine, in which, at each round, every contestant have the same probability to send a message (in the case of beeping model, message consists of one bit). These probabilities can be different for every time step, but have to be provided a priori (they do not depend on the history of past transmissions and receptions registered by a particular station). In Section 2.6 we provide, so called, Uniform Leader Election algorithm, which is independent from the idea of "uniform" arrangement of algorithm, described e.g. in [83]. Therefore in this dissertation, we rename the concepts from [83] as a *network-uniform* and a *network-non-uniform* algorithms. In the first of these two, at each time step, every agent sends a message with the same probability, which

[7] Otherwise we may specify a subgroup of such the devices with analogous assumptions restricted to this subgroup and perform Leader Election amongst them. All the other devices should can be eventually used as middlemen in communication between the competitors in the subgroup, especially when a subgraph induced by the subgroup is non-connected.

is dependent on the history of the evaluation. The idea of network-non-uniform algorithms additionally allows the devices to act dependent on theirs own history of communication. [8]

We also define a *quitting* algorithm, which allows the contestants to eventually power off (fall asleep) during the execution and remain in this state until the end of the procedure in order to save energy[9]. Otherwise, if the algorithm is network-uniform, then we call it *non-quitting*.

If LEA is deterministic (like e.g. in the aforementioned nameplate example), then there is no possibility for repetitive fairness. Also deterministic approach cannot ensure that LEA is totally reliable (see impossibility result from [93]. Therefore we only focus on randomized algorithms.

From the practical point of view, there are two universal types of randomized LEAs: *Las Vegas*[10] and *Atlantic City* algorithms[11]. The first one always either returns a unique leader or informs about a failure. However its runtime differs depending on the input and can be eventually very long and even unbounded. The latter one was originally defined as an algorithm, which has at most polynomial runtime (with respect to the network's size), but it can obtain uncertain result with probability at most 0.25. Nevertheless, in this dissertation, we curtail the notion, to at most logarithmically bounded runtime (with respect to the number of devices) and we restrain the algorithm to perform incorrectly with probability smaller than *mistake frequency* ε (given a priori). In general, when LEA successfully finds a leader with probability at least $1-\varepsilon$, then we say that it is $(1-\varepsilon)$-*reliable*. Note that each Atlantic City algorithm is inseparably connected with it mistake frequency and Las Vegas algorithm may or may not have such the parameter. Let us note, that when we consider quasi-anonymous $(1-\varepsilon)$-reliable algorithm, then we do not demand it to keep the reliability threshold, when the number of devices is bigger than the capacity of the network assumed a priori.

The previous partition of randomized LEAs is closely related to a termination problem. Randomized algorithms, which always return correct result but have *uncertain termination* (that terminates with probability close to 1) were widely considered before (see e.g. [89, 43]). Remark that these algorithms can be truncated to some number of steps (dependent on the initial parameters) and be treated as Atlantic City algorithms (probability of uncertain termination should be then included at a part of mistake frequency). Then we should have an idea how to properly choose a moment of artificial termination in order to guarantee demanded reliability. An algorithm, which utilize the truncated distribution is simply called a *restricted* one.

[8] Conversely to network-uniform algorithm, it does not force all the stations to perform in same way. Especially, some devices may be inactive in some rounds and may further wake-up to continue the process.

[9] For instance we can abuse the setup of oblivious or network-uniform in such the way, that the devices that are aware that are not more needed to obtain a leader may just turn off and wait for the procedure to end.

[10] Idea introduced by László Babai in 1979 in [5] as a dual approach to Monte-Carlo algorithms.

[11] Proposed in 1982 as a response to Las Vegas algorithms.

To sum up, in this dissertation we mainly provide universal, repetitively fair, oblivious-quitting, Atlantic City Leader Election algorithms with arbitrary small mistake frequencies, which are performed in single-channel, in fully-connected networks, in the arrangement of beeping model without collision detection. We will consider both non-anonymous and quasi-anonymous cases. We will also consider some other known solutions to compare the results.

2.2 A brief history of LEAs

In this section we present a short historical overview of Leader Election algorithms. An election of a leader is a fundamental problem in distributed systems and as we saw in Section 2.1, it is studied in a variety of contexts and scenarios.

A concept of Leader Election is often attributed to LeLann, who considered a problem of replacing a lost token in ring network in his paper from 1977 ([72]). He reformulated this problem in terms of Leader Election.

One of the most important early works on the problem of selection of a leader is the Dijkstra Prize-winning paper [50] from 1983, written by Gallager, Humblet and Spira. A core part of theirs solution is finding of a minimal-weight spanning tree of a network with disjoint weights of edges. Despite the fact that the Leader Election problem is a classic issue with many different solutions, especially when some specific topology of the network is given, an intensive research is still being carried out in this field.

In this section we focus on randomized solutions in clique topology, which can be easily generalized to algorithm appropriate for all connected topologies via extrema propagation technique (see e.g. [8, 9, 23]).

Therefore let us shortly discuss several examples of solutions for complete topology case from the literature.

Example 2. *At first, let us consider some popular variant of a very simple non-anonymous LEA which is, in fact, a core element of the Ethernet protocol from [78] from 1976, where each station sends some signal with probability* $p = \frac{1}{n}$. *Let* $\varepsilon > 0$ *be a mistake frequency,* K *be a number of synchronized rounds and consider the following procedure: each station* x *selects independently a sequence* $b_x = (x_1, \ldots, x_K)$ *of bits in such a way that* $\Pr[x_i = 1] = \frac{1}{n}$, *for every* $i \in [K]$. *Then* x *sends some signal at round* i *whenever* $x_i = 1$. *Let* $L_i = \operatorname{card}(\{x : x_i = 1\})$. *Notice that* $\Pr[L_i = 1] = \binom{n}{1}\frac{1}{n}\left(1 - \frac{1}{n}\right)^{n-1} \approx \frac{1}{e}$. *Let* S *denotes the event* $\bigvee_{i=1}^{K}(L_i = 1)$ *that exists a round in which a single station is audible (only one device is sending the signal). Then the first such the round marks a single agent, which can be designated as a leader. Note that* $\Pr[S] \approx 1 - \left(1 - \frac{1}{e}\right)^K$, *from which we can deduce that if*

$$K > \frac{\ln\left(\frac{1}{\varepsilon}\right)}{\ln\left(\frac{e}{e-1}\right)} \approx 2.18019\ldots \ln\left(\frac{1}{\varepsilon}\right) \ ,$$

then $\Pr[S] > 1 - \varepsilon$. *This observation can be transformed into a simple non-anonymous oblivious Atlantic City LEA, which requires* $\left\lceil 2.18019 \ln\left(\frac{1}{\varepsilon}\right)\right\rceil$ *rounds in a single-hop model that is* $(1-\varepsilon)$*-reliable.*

It worth to mention that in the above example the probability p of sending the signal is selected in such a way that $\Pr[S]$ is maximal (irrespective of K).

In 1993, H.Prodinger, in his paper titled "How to select a loser" [89], proposed a simple non-anonymous oblivious-quitting Las Vegas LEA via sequence of elimination rounds based on fair coin tossing. In this framework, the "loser" is equivalent to the nowadays leader. That nomenclature is quite intuitive, since it forces the agent who becomes a loser to work by flipping the coin. More detailed analysis was given by J.A. Fill, H. M. Mahmoud and W. Szpankowski in 1994 [43]. In [65, 76] LEAs based on biased coin tossing are concerned as well. We will present some of the results for fair coin case in Section 2.7.1 and provide a new contribution related to it, since Prodinger's algorithm is closely related to one of our contributions from Section 2.6.

In 1986, Dan Willard introduced two network-uniform Las Vegas LEAs. The first one in quasi-anonymity arrangement, which terminates in $\lg\lg N + \mathrm{O}(1)$ steps on average (where N is the network's capacity). The expected number of rounds of the second one is $\lg\lg n + \mathrm{o}(\lg\lg n)$ in fully-anonymity arrangement, where n is the number of all devices in the network. In years 1998-2002, K. Nakano and S. Olariu [81, 82, 83] proposed several fully-anonymous Las Vegas LEAs. One of them is network-non-uniform and terminates with probability $1-\varepsilon$ in at most $\lg\lg n + 2.28\lg\left(\frac{1}{\varepsilon}\right) + \mathrm{o}\left(\lg\lg n + \lg\left(\frac{1}{\varepsilon}\right)\right)$ time steps. Their another efficient solution is oblivious and terminates with probability $1-\varepsilon$ in at most $\mathrm{O}\left(\min\left((\lg n)^2 + \left(\lg\left(\frac{1}{\varepsilon}\right)\right)^2, \varepsilon^{-\frac{3}{5}}\lg n\right)\right)$ time steps. They also showed that Willard's second algorithm terminates with probability $1-\varepsilon$, for $\varepsilon \in \mathrm{O}(e^n)$, in at most $\lg\lg n + \Omega\left(\sqrt{\frac{1}{\varepsilon}}\right)$ rounds. Let us remark that these restrictions enable easy reformulations of the algorithms to theirs truncated, Atlantic City versions. An additional assumption about the network's capacity also allows to provide theirs quasi-anonymous versions. However, it worth to bear in mind that, by definition, oblivious algorithms are simpler that network-uniform and network-not-uniform algorithms. Let us note that in [83], the authors provided an illustrative comparison of theirs oblivious algorithm and second Willard's protocol. For instance, when $n = 10^6$, a Monte-Carlo experiment with 10^6 repetitions of the Willard's routine gave 9 runs which needed over 1000 rounds to correctly attain a leader and an appropriate test for the solution of Nakano and Olariu gave 14 runs with more than 40 rounds and maximally 52 time steps. It also worth to mention, that in 2013, in [51], Ghaffari et al. have considered an adaptation of Willard's fully-anonymous algorithm in multi-hop network.

Another interesting universal work has also been written in 2013 ([64]) by P. Jacquet, D. Milioris and P. Muhlethaler and concerns an energy efficient LEA which utilize random variable with geometric distribution. Due to its efficiency, the algorithm is often referred to as Leader Green Election (LGE). A precise description will be provided in Section 2.7.2 together with several propositions of

more efficient adaptations, since one of our main contributions from Section 2.8 uses the main idea from this paper, however attributed with several technical rectifications.

Next, well known solution, was proposed in 2015 by Y. Métivier, J.M. Robson and A. Zemmari in [79]. Theirs algorithms are called Splitting and Naming. They firstly splits the competitors in disjoint groups and further use random identity naming procedure in order to distinguish the identities inside the groups. The biggest identity in the group with the highest split-sign becomes the leader. Authors provided two methods to split and name the devices. Split-sign together with identity forms a label of the agent. Splitting and Naming are fully-anonymous, Las Vegas algorithms. The first proposed algorithm is successful "with high probability" (i.e. $1 - \mathrm{o}\left(n^{-1}\right)$; abbreviated to w.h.p.), the size of the labels is $\mathrm{O}\left(\lg(n)\right)$ w.h.p. and the expected value of this size is also $\mathrm{O}\left(\lg(n)\right)$. On the other side, the latter of proposed algorithms is successful "with very high probability" (i.e. $1 - \mathrm{o}\left(n^{-c}\right)$ for any $c \geqslant 1$; abbreviated to w.v.h.p.), the size of the labels is $\mathrm{O}\left(\lg(n)\lg^*(n)^2\right)$ w.v.h.p., however the expected value of labels' size is $\mathrm{O}\left(\lg(n)\lg^*(n)\right)$. Let us note, that one can restrict the splitting phase of the procedure in order to provide Atlantic City algorithm. According to the aforementioned properties, it is possible to terminate them in such the way that mistake frequency $\varepsilon(n)$ parameters asymptotically tend to 0 as the number of devices n tends to ∞. Let us remark that the Splitting and Naming arrangements are sometimes referred to as the state-of-the-art of Leader Election algorithms, however as we will see in Section 2.7.3, both solutions are unreliable for small number of devices and moreover, the second performs even worse in this case. This shows in particular, that they should not be used as quasi-anonymous Atlantic City algorithms.

2.3 Urn model

2.3.1 A description of a model

Commence with a description of a simple real life idea for LEA via tossing balls into labelled urns. We want to deal with this approach in details and also it has several meanings (for instance, in [76], a different model was proposed), hence this interpretation was postponed to separate section. One may recognize this model as a typical toy example, nevertheless the below description is very intuitive, yet quite general. Consider the following procedure:

- Assume that there are L urns, each signed with a unique number from the set $[L]$ (alternatively we may assume that labels come from some linearly ordered set of cardinality L, so especially we often also concern the set $[0 : L - 1]$).

- Imagine that there are n competitors: $v_1, \ldots, v_n$. Each competitor has a unique ball with its personal identifier (like e.g. ID from a nameplate or a signature) and tosses the ball into one of the urns according to some

rule given by *tossing generator*[12]. Tossing generator may be interpreted as a random number generator, which draws a number according to some distribution.

- The competitor whose ball is in the urn with the biggest number amongst the non-empty urns becomes a winner.

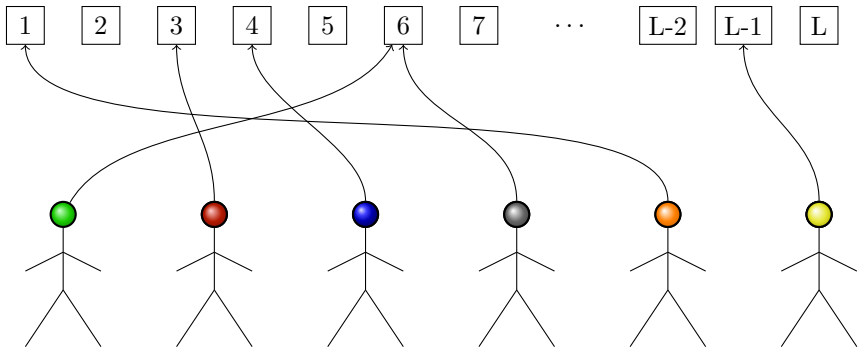

Figure 2.1: A schematic situation of Leader Election in urn model.

There are two possible results of such the arrangement. First of all, there may be exactly one winner — the leader. Then we say that the election is *successful*. This case is depicted in Figure 2.1, where a yellow agent chooses the urn $L-1$ and the rest competitors pick lower numbers. Otherwise, there are more winners and some additional steps are needed in order to achieve a unique leader. For instance, if we forget about the yellow agent from Figure 2.1, then there are two winners (green and grey), who choose 6-th urn. Urn model can be categorized as a kind of Atlantic City algorithm, which fails to provide unique leader, when there are multiple winners. Obviously we are interested in omitting the multiple winners case. Each such the situation is a *conflict* (since winners repeat the same value). When a conflict occurs, then the procedure may be reset and started in the same arrangement. However, it may be also wisely modified in order to reduce a chance of a next conflict or it may be extended to shortens the additional execution. Particularly, the procedure may be ran only by the winners of the previous instance of the procedure. This simple trick is utilized in several LEAs (e.g. in those presented in Section 2.7.2 and Section 2.7.3). Let us remark, that when we apply this idea to quitting algorithm, then after this manipulation, it remains quitting.

[12]When the rule is given a priori, then it provides oblivious algorithm and otherwise we deal either with network-uniform or network-non-uniform procedure.

2.3.2 A discussion about generality of urn model and its efficiency potential

We would like to heuristically justify that the urn model is the general approach to Leader Elections. Namely, we may think that each competitor draws a random property (the label). If the procedure is successful, one of the contestants becomes the leader, so he stands out in some sense — in other words he has "the best property". However "the best" implies an existence of some partial order on the set of possible properties, which defines the way we compare them. Note that every partial order $(X, \preceq)$ may be extended to some linear order $(X, \preceq^*)$ (see Section 1.2.3) that can be used instead in urn model. Then, for an arbitrary draw of properties, if $(X, \preceq)$ determines the leader, then so does $(X, \preceq^*)$. Hence, as long as we are interested in efficient approaches[13], we should use linear orders, so the one which arise in urn model.

Remark that urn model does not straightforwardly support a case, when each of the competitors choose some part of a property, which can be obtained, when all the chunks are known. However such the solution is unwieldy, because every contestant has to know all the parts, what can be memory consuming when the number of agents becomes big and it is also more complicated to execute in distributed systems or in anonymous networks, since every device need to get all the parts and know where they are from to omit duplicates. According to [93], "Every collective behaviour can be made homogeneous.", so one can always simplify the solution to symmetric and independent one. Namely, such the approach can be identified as drawing properties with some random partial order, that have to be established a posteriori, based on the randomness of tossing generators of each individual station. However such the solution is unnecessarily complicated, since in a case of class of fair algorithms in can be naturally substituted by one linear order.[14]

Example 3. *Consider the foregoing scenario of the election via urn model — in a group of competitors, there are at least two pretenders, who really want to win and they are capable to choose the urn of their will. Then, naturally, they always choose the urn with the biggest value. This scenario leads to nonsense — a never-ending conflict contest. Theoretically we may force somehow the algorithm to allow at most one of competitors to choose the urn with value L. However, such the idea is either unfair or the ball which is tossed to the urn with the sign L had to be chosen uniformly at random. Such the solution also requires a trusted curator, who is going to draw a winner, thus this curator act as a temporary leader. Hence this approach either fully simplifies the problem or leads to an absurd. These arguments show that we rather do not want LEAs*

[13] Here, efficient means that we would like to optimize the probability of success of LEA.

[14] Each agent uses initially one tossing generator. Every contestant can instead choose uniformly a tossing generator amongst all used by competitors (with eventual repetitions if the same independent tossing generator is used by more than one device). Then, for every partial order, each node has the same probability of win, so the procedure remains repetitively fair. Also the linear order can be established in such the way to maximize the probability of success.

to be deterministic or dependent on some outer decision-makers.

In fair LEA, all the competitors are equally likely to win. A simple and very natural solution is to let them put the balls according to the same distribution over the set of urns (e.g. numbered by elements of $[L]$). Nevertheless there is also a possibility of an arrangement, which allow the competitors to choose different distributions and still remain fair.

Example 4. *Assume that there are two participants of LEA in urn model on* $[3]$*, where the first competitor uses the uniform distribution* $\mathrm{Uni}\,(3)$ *and the second uses a tossing generator with simple probability mass function:* $\Pr[X = 1] = \Pr[X = 3] = \frac{1}{2}$.
Then a tie in urn model occurs with probability $2 \cdot \frac{1}{2} \cdot \frac{1}{3} = \frac{1}{3}$. *The second participant wins with probability* $\frac{1}{2} \cdot \frac{2}{3} = \frac{1}{3}$*, hence the first one also wins with the same probability, so such the LEA is repetitively fair.*

At the first glance such the solution seems all right. However, in practise, this type of approach is a little bit problematic. Imagine that the leader was established by similar method, but he disconnects from a network and a new leader has to be chosen. Could competitors use the same tossing generator as before and preserve the fairness? A next example shows that sometimes in such the model, the probability distributions should be rearranged, what is a serious snag, especially when the network is anonymous:

Example 5. *Assume that there are* 3 *competitors of LEA, each with different probability mass functions, given by the vectors:* $D_1 = (0, 1, 0), D_2 = \left(\frac{2}{3}, 0, \frac{1}{3}\right)$ *and* $D_3 = \left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$*. It is easy to check that every contestant has the same probability of win in the urn model —* $\frac{2}{9}$.
Imagine that the third competitor suddenly disconnects. Then the first competitor now has a chance of win – $\frac{2}{3}$ *– and the second one –* $\frac{1}{3}$ *– so the fairness is not preserved.*

Note that it is not known how to change the distributions when a new competitor appears or some group of the competitors withdraw. Moreover, if some of competitors resigns during the execution of LEA, then the relative chances of election of the rest of the active competitors can change. A similar situation can happen when a new device connects to the network and the present leader decides to perform a reelection. For this reason this asymmetric arrangement is perplexing. From now on we will assume that the procedure is fully symmetric, so in particular all the competitors use the same probability distribution and Leader Elections are repetitively fair.

To sum up, we say that LEA follows the urn model if it chooses a winner according to beneath rules:

- Every user in the group use tossing generator, which independently draws some identity according to the same discrete distribution $\bar{p}^{(L)}$ over $[L]$[15],

[15]We eventually allow the distribution $\bar{p}^{(\infty)}$ to be defined on $\mathbb{N}$, which is the countable set contrary to $[L]$. This variant is especially important, when one want to obtain reliable fully-anonymous algorithm.

- A user with the biggest identity becomes the group leader,
- If at least 2 of the competitors draw the same maximal identity amongst all contestants, then a conflict occurs.

2.4 General block of Leader Election algorithm

Consider the following simple, yet quite general, restricted Leader Election pseudo-code:

ALGORITHM 1: Base Leader Election Algorithm Block

```
   procedure Select(D,Θ,L)                                  // For each node
1  |  generate X ∼ D(Θ)
2  |  broadcast m := min(X, L)
3  |  calculate maximum value M of all messages send by all nodes
4  |  Leader := [[m = M]]
```

An input of Algorithm 1 consists of three parameters: a probability distribution D ranged in $\mathbb{N}$, a set of distribution parameters Θ of the distribution D and a threshold $L \in \mathbb{N} \cup \{\infty\}$. First of all, each process generates a random variable X according to the distribution $D(\Theta)$. Further each of devices restrict the obtained realization of X to the set $[L]$, so every value exceeding L is therefore substituted by L.[16] Note that one can initially assume that D has a support constrained to $[L]$ and that Algorithm 1 is compliant with the urn model with L urns. During the third step, all the devices communicate the attained restricted identifiers to all the other nodes and establish the maximal value M among them. Note that this part mainly depends on a topology of the network and a model of transmission. All the processes which own the restricted identifier that equals to M, become leaders. This latter fragment of the algorithm may be completed with the extrema propagation methods mentioned in Section 2.1, which inform all the other processes about the present leader.

A natural goal is to search for the parameters D, Θ and L, for which LEA would be successful with reasonable probability and for the communication model in which these parameters guarantee possibly simple transmission. In order to fulfil the first requirement, we demand the algorithm to be $(1 - \varepsilon)$-reliable. The second condition can be attained by a minimization of L parameter, so the complexity of the longest possible message (containing identifier) is maximally reduced and therefore the runtime of the algorithm as well. Let us denote $K := \lceil \lg(L) \rceil$ to be a *time cost*. We further justify such the designation. Note that since this protocol is distributed, we mainly focus on the maximum of costs taken by the devices, not the sum of these outlays. Nevertheless, the second yardstick is also very interesting and will be referred to as *total energy cost*. Realize that both measures used for optimizations are closely related to a

[16] Note that we allow the algorithm to have senseless constraint, when $L = \infty$. It is for the purpose of fully-anonymous algorithms.

problem of minimization of energy consumption of the contestants. Indeed, the longer is the message, the more energy has to be used to sent it.

2.4.1 Beeping model, single-hop and multi-hop arrangements

In practise, messages can be sent in very simple, so called, *beeping model* (introduced by Cornejo and Kuhn in [28]). In this paradigm of communication, an execution of a distributed algorithm is divided into synchronized disjoint rounds. Every time step, each process can either send a BEEP signal to its neighbours or listen to a transmission channel (receive BEEPs from its neighbours).[17] It worth to mention that in practise each device may differ, and so can a setup of broadcasting parameters. For example, beeping can have different frequencies or amplitudes of power. Therefore when a node hears more than a single BEEP at once (during one round) when listening, then he can potentially distinguish such a signal from a single BEEP. Hence a case of multiple BEEPs is called a *collision* and if devices can distinguish a single BEEP from a collision, then we say about beeping model with *collision detection*.

If a graph of topology of a network is complete, then the network is in a *single-hop* arrangement (every communication between any two devices in the network can be done with a single step/hop). Otherwise, when some of the transmissions between devices has to be bypassed, we say that the network has a *multi-hop* arrangement.

In single-hop arrangement, we can use beeping model and a method from [64] to provide the foregoing algorithm:

ALGORITHM 2: Beeping Model of Communication designated for Leader Election in Urn Model

```
   procedure Send(D, Θ, L)                                  // For each node
1  |  Leader = true
2  |  generate X ~ D(Θ)
3  |  msg = min(X, L)
4  |  K = ⌈lg(L)⌉
5  |  B = BIN_K(msg − 1)
6  |  for i = K − 1 down to 0 do
7  |  |  if [[B[i] = 1]] then
8  |  |  |  send BEEP
9  |  |  else
10 |  |  |  listen
11 |  |  |  if hear BEEP or collision then        // other node transmits
12 |  |  |  |  Leader = false
13 |  |  |  |  break
```

[17] Since we have assumed (see Section 2.1) that the communication is performed via single-channel, the messages are heard by all the neighbours, which are listening to and are in the range of transmission.

Algorithm 2 is an adaptation of Algorithm 1 to beeping model in single-channel single-hop arrangement. A message msg corresponds to m from Algorithm 1. We would like to represent it in binary form, so it is better to subtract 1 from it or alternatively consider distributions D on $[0 : L-1]$ instead[18]. K is the minimal number of bits needed to transmit msg and B is a binary form of msg -1. Remark that $\text{msg} - 1 = \sum_{i=0}^{K-1} B_i 2^i$. Therefore $\sum_{i=0}^{K-1} M_i 2^i$, where M_i is maximal B_i amongst the competitors, is exactly M in terms of Algorithm 1. Note that in Algorithm 2, if some device hears BEEP or collision at some round i, then its $B[K-i] = 0$ and there is at least one agent which sends BEEP in that round. Then all senders has $B[K-i] = 1$. According to the observation about M, all those active listeners do not have maximal msg, so they cannot be leader. Therefore they sets $Leader =$ **false**, interrupt the transmission and resigns (lines 11-13 of Algorithm 2). Realize that the termination can be realized as the sleeping phase of quitting algorithm in order to save energy.[19] In order to perform a correct Leader Election, all the contestants should initially believe that they are the potential winners, so in line 1 we set $Leader =$ **true**.

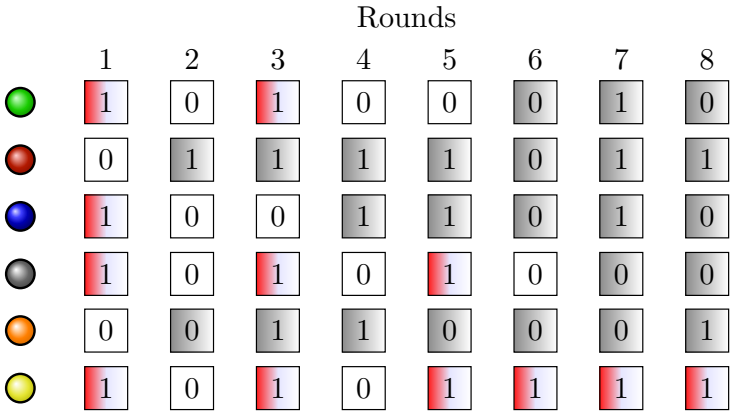

Figure 2.2: An example of beeping model of communication for Leader Election in urn model for 6 devices (colorful balls) in complete network. Devices previously generated messages (msg): 163, 124, 155, 169, 50 and 176 respectively (in order from top to bottom).

In Figure 2.2 we present an illustration of exemplary LEA performed by 6 agents according to Algorithm 2, where msg -1 are drawn independently from the distribution $\text{Uni}\left([0 : 2^8 - 1]\right)$ ($K = 8$). Red squares denote BEEP and grey indicate inactive nodes (those which terminated the communication). Let

[18] However then one should care about slight changes in definitions of K and B. Whenever we use this shifted approach in next sections, we indicate it manifestly.

[19] Note that every device know the number of the time steps that remain until the end of the algorithm.

us consider the presented process for a while. Two devices (red and orange) resign during the first round, since they hear that other nodes BEEP (they have drawn msg $\leqslant 128$). Therefore the red agent did not transmit in the second round although its $B_6 = 1$. All the other devices has $B_6 = 0$, so nobody BEEPs in the second round. Note that the winner (yellow ball) has to transmit till the end, although all the others are inactive since the 7-th round. One can also see that the leader BEEPs 6 times in the example. This is also the total number of BEEPs of all other devices. It worth to realize that the lower is msg of a device, the faster it falls asleep (on average).

Note that Algorithm 2 can be carried out even in more general model without collision detection. We sometimes refer to K as an *efficiency rate* (also alternatively *time complexity* or *time cost*) of Algorithm 1.

An algorithm which takes some redundant rounds may be interpreted as a software bloat. This affirms our decision to optimize the efficiency rate of LEAs.

In the case of multi-hop network one may use an emulation of an algorithm in single-hop arrangement, performed in multi-hop one ([10]) or the well-known extrema propagation algorithm (presented in [8, 9, 23]). Let us only mention that these algorithms are directly proportional with respect to a diameter of the network. Therefore Algorithm 1 can be also adapted to multi-hop environment. However we omit a full description of this procedure, since it is not the main topic of our interest.

2.4.2 Probability of Choosing Unique Maximal Element

Let us assume that there are n competitors, which use Algorithm 1 in order to obtain a leader. Let $(X_i)_{i\in[n]}$ be a sequence of i.i.d. random variables with a distribution ranged in some linearly ordered set $(\Omega, \preceq)$. Hereinafter, we interpret that each random variable X_i selects an urn behalf the competitor denoted by number i. We would like to establish a probability of successful LEA (i.e. there exists only one competitor which selects the element $\max_{\preceq}\{X_i : i \in [n]\}$). We start with a general result about this probability. Specifically Ω can be a subset of $\mathbb{N}$ and $\preceq$ is a linear order defined on Ω (e.g. $\leqslant$ in the case of subsets of $\mathbb{N}$).

Theorem 2.4.1. *Let $n > 1$ and $X_1, \ldots, X_n$ be a sequence of i.i.d. random variables with values in the set $\mathbb{N}$. Let S denotes the event that there is a unique maximum amongst $X_1, \ldots, X_n$:*

$$(\exists\, i \in [n])(\forall\, j \in [n]\backslash\{i\})\ (X_j < X_i)\ .$$

Then

$$\Pr[S] = n \sum_{k=2}^{\infty} \Pr[X_1 = k] \Pr[X_1 < k]^{n-1}\ . \tag{2.1}$$

Before a proof, let us remark several observations:

- If $(X_i)_{i\in[n]}$ are random i.i.d. identities of competitors used to establish a leader by choosing the one with the biggest identity, then S can be

interpreted as the event that the algorithm returns a unique leader (i.e. the procedure is successful),

- Every agent is equally likely to become the leader,
- The leader has to draw at least value 2,
- All the others has to draw smaller numbers.

Note that Theorem 2.4.1 allows the set of urns to be countable. However it can be easily truncated to a finite number of urns by setting 0 for the probability of tossing a ball into the urns with labels exceeding some threshold.

Proof. The random variables $X_1, \ldots, X_n$ are identically distributed, so

$$\Pr[(\forall\, j \neq a)\ (X_j < X_a)] = \Pr[(\forall\, j \neq b)\ (X_j < X_b)]$$

for all $a, b \in [n]$. Hence $\Pr[S] = n\Pr\left[\bigwedge_{j=2}^{n} (X_j < X_1)\right]$. Therefore

$$\Pr[S] = n\sum_{k=1}^{\infty} \Pr\left[\bigwedge_{j=2}^{n} (X_j < X_1) \middle| X_1 = k\right] \Pr[X_1 = k] =$$

$$n\sum_{k=1}^{\infty} \Pr\left[\bigwedge_{j=2}^{n} (X_j < k)\right] \Pr[X_1 = k] = n\sum_{k=2}^{\infty} \Pr[X_1 < k]^{n-1} \Pr[X_1 = k]\ .$$

□

Note that general formula for the success probability of Leader Election algorithm in urn model is given by Eq. (2.1). It is difficult to analyze the properties of this formula in its basic form for an arbitrary distribution. However for some probability distributions, which are important for applications, we can use (2.1) and derive specific formulas which can be approximated with required precision.

2.5 Non-anonymous Leader Election Algorithm

2.5.1 A description of a problem and definitions

We assume that there exists a group of n devices and we desire to provide a group leader by a repetitively fair and non-anonymous Leader Election (abbreviated to NALEA) in urn model. Basing on the dispute from Section 2.4.2, we may identify each person with a random variable.

Let $n \in \mathbb{N}$ and $X, X_1, X_2, \ldots, X_n$ be i.i.d. random variables ranged in $[L]$ (note that we technically allow the distribution to have some null atoms, so it has to be taken into considerations). X_i corresponds to an identity of i-th

person in the group, for $i \in [n]$. Moreover, let ${p_j}^{(L)} := \Pr[X = j]$ for $j \in [L]$. Consider an event S_n that there exists exactly one maximum in a multi-set $Z_n := \mathrm{MSet}\{x_1, x_2, \ldots, x_n\}$, where x_i is a realization of the variable X_i (with $i \in [L]$).[20] In the context from Section 2.1 and Section 2.3, we may alternatively interpret S_n as the success of Leader Election algorithm, based on Algorithm 2 with $D(\Theta) = \bar{p}^{(L)}$, in urn model and this event can be defined directly by the formula: $S_n := [\![\mathrm{card}\,(x_i : x_i = \max(Z_n)) = 1]\!]$. We will consider how $\Pr[S_n]$ depends both on n and L, so we are going to denote this relationship explicitly by $\Pr\big[\bar{p}^{(L)}, n\big]$ instead.

We would like to provide reliable algorithms, hence our goal is to optimize $\Pr\big[\bar{p}^{(L)}, n\big]$ for any given n and L. More precisely, for fixed n and L, we are searching for such the distribution (denoted by $\bar{p}(n)^{(L)}$), which maximizes the success probability $\Pr\big[\bar{p}^{(L)}, n\big]$. In Section 2.5.2 we will show that such the distribution is unique.

2.5.2 Optimization of the Probability of Success

Realize that for any $n \geqslant 2$, we may rewrite formula (2.1) from Theorem 2.4.1 in this case in the following way:

$$\Pr\Big[\bar{p}^{(L)}, n\Big] = n \sum_{i=2}^{L} {p_i}^{(L)} \left(\sum_{k=1}^{i-1} {p_k}^{(L)} \right)^{n-1} . \tag{2.2}$$

Remark that the function (2.2) is continuous in Euclidean topology on $\mathbb{R}^L$ and takes values from the interval $[0, 1]$.

By Weierstrass Extreme Value Theorem (Theorem 1.4.5), $\Pr\big[\bar{p}^{(L)}, n\big]$ reaches its minimum and maximum in Sim_L (see Section 1.2 for definition). A boundary of Sim_L consists of all the distributions, for which there exists some $j \in [L]$ such that ${p_j}^{(L)} = 0$. However, when it happens, then we can think of it as a distribution on $[L-1]$, because we can shift all the atoms ${p_i}^{(L)}$ for $i > j$ by one index. Hence, without a loss of generality, we may assume that ${p_L}^{(L)} = 0$. Since there may be more than one null atoms, we may assume that there exists such the $m < L$ that ${p_i}^{(L)} > 0$ for $i \leqslant m$ and ${p_i}^{(L)} = 0$ for $i > m$. We will prove that such the degenerated distribution cannot maximize the probability of success:

Lemma 2. *Let $n \geqslant 2$ and $m, L \in \mathbb{N}$ be fixed and $m < L$. Moreover, let $\bar{p}^{(L)} \in \mathrm{Sim}_L$ be such that ${p_i}^{(L)} = 0$ for $i > m$. Then there exists $\bar{p'}^{(L)} \in \mathrm{Sim}_L$ such that $\Pr\big[\bar{p}^{(L)}, n\big] < \Pr\big[\bar{p'}^{(L)}, n\big]$.*

Proof. Assume that $\bar{p}^{(L)}$ maximizes $\Pr\big[\bar{p}^{(L)}, n\big]$. We show that then exists a distribution $\bar{p'}^{(L)}$ which has higher probability of success. Let us define ${p'_i}^{(L)} = {p_i}^{(L)}$ for $i \leqslant m-1$ and ${p'_m}^{(L)} = \frac{1}{2}{p_m}^{(L)} = {p'_{m+1}}^{(L)}$. Moreover, $p'_i = 0$ for

[20] Multi-set is a collection of elements, which allows multiple instances of the same element in contrast to a set.

$i \in [m+2 : L]$. Then by Eq. (2.2), we get

$$\Pr\left[\bar{p}^{(L)}, n\right] = n \sum_{i=2}^{m} {p_i}^{(L)} \left(\sum_{k=1}^{i-1} {p_k}^{(L)} \right)^{n-1} ,$$

so by the definition of $\bar{p'}^{(L)}$, we have

$$\Pr\left[\bar{p'}^{(L)}, n\right] = n \sum_{i=2}^{m-1} {p_i}^{(L)} \left(\sum_{k=1}^{i-1} {p_k}^{(L)} \right)^{n-1} + \frac{n}{2} {p_m}^{(L)} \left(\sum_{k=1}^{m-1} {p_k}^{(L)} \right)^{n-1} + \\ + \frac{n}{2} {p_m}^{(L)} \left(\sum_{k=1}^{m-1} {p_k}^{(L)} + \frac{1}{2} {p_m}^{(L)} \right)^{n-1} .$$

The comparison of two probabilities attained above bears

$$\Pr\left[\bar{p}^{(L)}, n\right] - \Pr\left[\bar{p'}^{(L)}, n\right] = \frac{n}{2} {p_m}^{(L)} \left(\sum_{k=1}^{m-1} {p_k}^{(L)} \right)^{n-1} + \\ - \frac{n}{2} {p_m}^{(L)} \left(\sum_{k=1}^{m-1} {p_k}^{(L)} + \frac{1}{2} {p_m}^{(L)} \right)^{n-1} < 0 .$$

□

We have just proved that the distribution which maximize the probability of success has the biggest possible support $[L]$, or in other words, the probability of success is maximized in an interior of Sim_L.

Theorem 2.5.1. *Let $n \geqslant 2$ and $L \in \mathbb{N}$ be fixed. There exists a unique distribution $\bar{p}^{(L)} \in \mathrm{Sim}_L$, which maximizes the probability of success of non-anonymous Leader Election in urn model for n competitors. This distribution satisfies two recursive relations, for any $j \in [L]$:*

$$p_j(n)^{(L)} = \frac{\frac{\left(\sum_{k=1}^{j-1} p_k(n)^{(L)} \right)^{n-1}}{n-1} - \sum_{i=2}^{j-1} p_i(n)^{(L)} \left(\sum_{k=1}^{i-1} p_k(n)^{(L)} \right)^{n-2}}{\left(\sum_{k=1}^{j-1} p_k(n)^{(L)} \right)^{n-2}}$$

and

$$p_j(n)^{(L+1)} = p_j(n)^{(L)} \cdot \left(1 - p_{L+1}(n)^{(L+1)}\right) .$$

Lemma 3. *There exists a unique distribution $\bar{p}^{(L)}$, which maximizes (2.2), i.e.*

$$n \sum_{i=2}^{L} {p_i}^{(L)} \left(\sum_{k=1}^{i-1} {p_k}^{(L)} \right)^{n-1} .$$

Proof. The proof utilizes the method of Lagrange multipliers (see e.g.[13] for details of the method). Obviously all atoms of any discrete random variable sum up to 1, so from (2.2), for any $\lambda \in \mathbb{R}$:

$$\Pr\left[\bar{p}^{(L)}, n\right] = n \sum_{i=2}^{L} {p_i}^{(L)} \left(\sum_{k=1}^{i-1} {p_k}^{(L)} \right)^{n-1} - \lambda \left(\sum_{i=1}^{L} {p_i}^{(L)} - 1 \right) .$$

In order to find extrema of the above function, we would like to check the signs of derivatives of $\Pr\left[\bar{p}^{(L)}, n\right]$ with respect to all ${p_j}^{(L)}$ variables. For any $j \in [L]$, we obtain

$$\frac{\partial \Pr\left[\bar{p}^{(L)}, n\right]}{\partial p_j} = n(n-1) \sum_{i=j+1}^{L} {p_i}^{(L)} \left(\sum_{k=1}^{i-1} {p_k}^{(L)} \right)^{n-2} - \lambda + n \left(\sum_{k=1}^{j-1} {p_k}^{(L)} \right)^{n-1} .$$

Note that, when $j = 1$, then the last summand vanishes.

If all the derivatives of the first order are zeros, then both sides of the beneath formula equal to $\frac{\lambda}{n(n-1)}$ (for any $j \in [L]$):

$$\sum_{i=2}^{L} {p_i}^{(L)} \left(\sum_{k=1}^{i-1} {p_k}^{(L)} \right)^{n-2} = \sum_{i=j+1}^{L} {p_i}^{(L)} \left(\sum_{k=1}^{i-1} {p_k}^{(L)} \right)^{n-2} + \frac{\left(\sum_{k=1}^{j-1} {p_k}^{(L)} \right)^{n-1}}{n-1} .$$

After a subtraction of a common part of the above equation we obtain:

$$(n-1) \sum_{i=2}^{j} {p_i}^{(L)} \left(\sum_{k=1}^{i-1} {p_k}^{(L)} \right)^{n-2} = \left(\sum_{k=1}^{j-1} {p_k}^{(L)} \right)^{n-1} \tag{2.3}$$

for $j \in [2 : L]$. Note that $\Pr\left[\bar{p}^{(L)}, n\right]$ is a polynomial function of the coordinates of $\bar{p}^{(L)}$, hence it is continuous. It is defined on the simplex Sim_L, which is closed and bounded set in $\mathbb{R}^L$, hence it is compact as well. By Weierstrass Extreme Value Theorem 1.4.5, $\Pr\left[\bar{p}^{(L)}, n\right]$ reaches its maximum and minimum in Sim_L. However, Lemma 2 shows that on the boundary of the simplex, the polynomial (2.2) does not achieve maximum, so it is realized in the stationary point in an interior of the simplex.

Now, realize that in Eq. (2.3), the term ${p_j}^{(L)}$ appears only once, on the left hand side. The rest depends only on atoms ${p_i}^{(L)}$, for $i < j$, and the number of competitors n. Therefore, for $j \geqslant 2$, the stationary point satisfies:

$$p_j(n)^{(L)} = \frac{\frac{\left(\sum_{k=1}^{j-1} p_k(n)^{(L)} \right)^{n-1}}{n-1} - \sum_{i=2}^{j-1} p_i(n)^{(L)} \left(\sum_{k=1}^{i-1} p_k(n)^{(L)} \right)^{n-2}}{\left(\sum_{k=1}^{j-1} p_k(n)^{(L)} \right)^{n-2}} \tag{2.4}$$

together with the boundary condition

$$\sum_{i=1}^{L} p_i(n)^{(L)} = 1 \ . \tag{2.5}$$

From Eq. (2.4), by simple transformations, we can find the dependence between $p_j(n)^{(L)}$ and $p_1(n)^{(L)}$, for any $j \in [2 : L]$. Namely, let us denote $a_j(n)^{(L)} = \frac{p_j(n)^{(L)}}{p_1(n)^{(L)}}$. Note that for any $j \in [2 : L]$, both numerator and denominator of the right hand side of Eq. (2.4) are polynomials in variables $p_i(n)^{(L)}$ (where $i \in [j]$) of degrees $n-1$ and $n-2$ respectively and in both cases, there are no monomials of lower degrees. Therefore, we can omit (L) in superscript of $a_j(n)^{(L)}$, because it is not dependent on L parameter, and consequently Eq. (2.5) can be rewritten to a form $C(n)^{(L)} \cdot p_1(n)^{(L)} = 1$, where $C(n)^{(L)} := \sum_{i=1}^{L} a_i(n)$.[21] Therefore $p_1(n)^{(L)} = \frac{1}{C(n)^{(L)}}$ and any $p_j(n)^{(L)}$ (for $j \in [2 : L]$) can be obtained recursively from Eq. (2.4) as well. This shows that the stationary point is unique, so it also has to maximize the probability of success from Eq. (2.2). □

Note that $C(n)^{(L)}$ from the proof of Lemma 3 is explicitly dependent only on n, but in fact, $C(n)^{(L)}$ is also dependent on L, since it arise as a sum of L terms $a_i(n)$.

The following lemma justifies the linear relation between atoms $p_j(n)^{(L)}$ and $p_1(n)^{(L)}$ (for $j \in [2 : L]$) and suggests a way by which $C(n)^{(L)}$ can be computed:

Lemma 4. *For any $n \in \mathbb{N}$, there exists a sequence of functions $(a_i(n))_{i\in\mathbb{N}}$ such that for any $L \in \mathbb{N}$ and $j \in [L]$*

$$p_j(n)^{(L)} = a_j(n) p_1(n)^{(L)} \ . \tag{2.6}$$

Moreover, $a_1(n) = 1$ and, for $j \in [2 : L]$, $(a_i(n))_i$ satisfies the foregoing recursion:

$$a_j(n) = \frac{\sum_{k=1}^{j-1} a_k(n)}{n-1} - \frac{\sum_{i=2}^{j-1} a_i(n) \left(\sum_{k=1}^{i-1} a_k(n)\right)^{n-2}}{\left(\sum_{i=1}^{j-1} a_i(n)\right)^{n-2}} \ . \tag{2.7}$$

Proof. First of all, realize that $a_1(n) = 1$ simply fulfills (2.6).
Assume that (2.7) and (2.6) are satisfied for all $k \leqslant j$, for some $j \in [L-1]$.

[21] It is formally proved in Lemma 4

Then from (2.4) we attain

$$p_{j+1}(n)^{(L)} = \frac{\frac{\left(\sum_{k=1}^{j} p_k(n)^{(L)}\right)^{n-1}}{n-1} - \sum_{i=2}^{j} p_i(n)^{(L)} \left(\sum_{k=1}^{i-1} p_k(n)^{(L)}\right)^{n-2}}{\left(\sum_{k=1}^{j} p_k(n)^{(L)}\right)^{n-2}}$$

$$= \frac{\frac{\left(\sum_{k=1}^{j} a_k(n)p_1(n)^{(L)}\right)^{n-1}}{n-1} - \sum_{i=2}^{j} a_i(n)p_1(n)^{(L)} \left(\sum_{k=1}^{i-1} a_k(n)p_1(n)^{(L)}\right)^{n-2}}{\left(\sum_{k=1}^{j} a_k(n)p_1(n)^{(L)}\right)^{n-2}}$$

$$= \frac{\sum_{k=1}^{j} a_k(n)p_1(n)^{(L)}}{n-1} - \frac{\sum_{i=2}^{j} a_i(n) \left(\sum_{k=1}^{i-1} a_k(n)\right)^{n-2} \cdot p_1(n)^{(L)}}{\left(\sum_{k=1}^{j} a_k(n)\right)^{n-2}} .$$

□

Note that according to Eq. (2.7), we can simply calculate that $a_2(n) = \frac{1}{n-1}$.

Remark that the relation (2.7) between $p_j(n)^{(L)}$ and $p_1(n)^{(L)}$ does not depend on L (it depends on j), so for any given $n \in \mathbb{N}$, the sequence $(a_i(n))_{i\in\mathbb{N}}$ is universal. The foregoing lemma justifies this observation:

Lemma 5. *Fix $L, n \in \mathbb{N}$ and let $i \in [L]$. Then*

$$p_i(n)^{(L+1)} = p_i(n)^{(L)} \cdot (1 - p_{L+1}(n)^{(L+1)}) . \tag{2.8}$$

Proof. From (2.5) and Lemma 4 we have

$$1 = \sum_{i=1}^{L} p_i(n)^{(L)} = \sum_{i=1}^{L} a_i(n)p_1(n)^{(L)} ,$$

thence, for any $n, L \in \mathbb{N}$, we get

$$p_1(n)^{(L)} = \left(\sum_{i=1}^{L} a_i(n)\right)^{-1} . \tag{2.9}$$

From (2.6) we attain $p_i(n)^{(L)} = a_i(n)p_1(n)^{(L)}$ and

$$p_i(n)^{(L+1)} = a_i(n)p_1(n)^{(L+1)} = p_i(n)^{(L)} \frac{p_1(n)^{(L+1)}}{p_1(n)^{(L)}} .$$

However, thanks to (2.9) we obtain

$$\frac{p_1(n)^{(L+1)}}{p_1(n)^{(L)}} \overset{(2.9)}{=} \frac{\sum_{i=1}^{L} a_i(n)}{\sum_{i=1}^{L+1} a_i(n)} = 1 - \frac{a_{L+1}(n)}{\sum_{i=1}^{L+1} a_i(n)} = 1 - \frac{a_{L+1}(n)p_1(n)^{(L)}}{\sum_{i=1}^{L+1} a_i(n)p_1(n)^{(L)}}$$
$$\overset{(2.6)}{=} 1 - \frac{{p_{L+1}}^{(L+1)}}{\sum_{i=1}^{L+1} p_i(n)^{(L+1)}} \overset{(2.5)}{=} 1 - {p_{L+1}}^{(L+1)} ,$$

what ends the proof. □

Lemmas 3, 4 and 5 together prove Theorem 2.5.1.

Lemma 5 shows that, for every $n \in \mathbb{N}\backslash\{1\}$, the sequence $(a_i(n))_{i\in\mathbb{N}}$ is decreasing. Realize that (2.6), (2.7) and (2.9) combined together enable us to provide the optimal distribution explicitly. We present some simple examples of such the distributions in Appendix A.

2.5.3 Approach via probabilities of non-last urns

Assume that we use urn model of Leader Election, according to probability $\bar{p}(n)^{(L)}$, obtained in Section 2.5.2. Then a probability that a single ball is not tossed into the last amongst L urns is $1 - p_L(n)^{(L)}$. Since Lemma 5 showed the importance of this term, let us denote it by $q_L(n)$.

We are going to show surprisingly compact recursive relation of first order satisfied by sequences $(q_L(n))_L$:

Theorem 2.5.2. *Let $p(n)^{(L)} \in \mathrm{Sim}_L$ be optimal distribution for n competitors for the problem of non-anonymous Leader Election in urn model and $q_L(n) = 1 - p_L(n)^{(L)}$. Then*

$$q_{L+1}(n) = \frac{n-1}{n - q_L(n)^{n-1}} \tag{2.10}$$

with the initial condition $q_1(n) \equiv 0$.

Proof.

$$q_{L+1}(n) \overset{(2.4)}{=} 1 - \frac{\frac{\left(\sum_{k=1}^{L} p_k(n)^{(L+1)}\right)^{n-1}}{n-1} - \sum_{i=2}^{L} p_i(n)^{(L+1)} \left(\sum_{k=1}^{i-1} p_k(n)^{(L+1)}\right)^{n-2}}{\left(\sum_{k=1}^{L} p_k\,(n)^{\,(L+1)}\right)^{n-2}}$$

$$\begin{aligned}
\dots &\overset{(2.8)}{=} 1 - \frac{\sum_{k=1}^{L} p_k(n)^{(L)} \cdot q_{L+1}(n)}{n-1} \\
&+ \frac{\sum_{i=2}^{L} p_i(n)^{(L)} \cdot q_{L+1}(n) \left(\sum_{k=1}^{i-1} p_k(n)^{(L)} \cdot q_{L+1}(n) \right)^{n-2}}{\left(\sum_{k=1}^{L} p_k\,(n)^{\,(L)} \cdot q_{L+1}(n) \right)^{n-2}} \\
&= 1 - q_{L+1}(n) \left(\frac{\sum_{k=1}^{L} p_k(n)^{(L)}}{n-1} - \frac{\sum_{i=2}^{L} p_i(n)^{(L)} \left(\sum_{k=1}^{i-1} p_k(n)^{(L)} \right)^{n-2}}{\left(\sum_{k=1}^{L} p_k(n)^{(L)} \right)^{n-2}} \right) \\
&\overset{(2.5)}{=} 1 - q_{L+1}(n) \left(\frac{1}{n-1} - \sum_{i=2}^{L} p_i(n)^{(L)} \left(\sum_{k=1}^{i-1} p_k(n)^{(L)} \right)^{n-2} \right)
\end{aligned}$$

By comparing the marginal expressions of the above chain of transformations we attain:

$$q_{L+1}(n) = \frac{n-1}{n - (n-1) \sum_{i=2}^{L} p_i(n)^{(L)} \left(\sum_{k=1}^{i-1} p_k(n)^{(L)} \right)^{n-2}} \, .$$

Realize that for $n > 2$ the above formula can be alternatively expressed as

$$q_{L+1}(n) = \frac{n-1}{n - \Pr\left[\bar{p}(n)^{(L)}, n-1\right]} \, . \tag{2.11}$$

Let us note that naturally, in (2.11),

$$\Pr\left[\bar{p}(2)^{(L)}, 1\right] = 1 \neq 1 - p_1(2) = \sum_{i=2}^{L} p_i(2)^{(L)} \left(\sum_{k=1}^{i-1} p_k(2)^{(L)} \right)^{0} \, .$$

Nevertheless, hereinafter we interpret $\Pr\left[\bar{p}(n)^{(L)}, n-1\right]$ formally as

$$(n-1) \sum_{i=2}^{L} p_i(n)^{(L)} \left(\sum_{k=1}^{i-1} p_k(n)^{(L)} \right)^{n-2} \, ,$$

even for $n = 2$.

For $j \in [L]$, let us denote the j-th partial sum of $\Pr\left[\bar{p}(n)^{(L)}, b\right]$ by the expression $\Pr\left[\bar{p}(n)^{(L)}, b, j\right]$, i.e.

$$\Pr\left[\bar{p}(n)^{(L)}, b, j\right] := b \sum_{i=2}^{j} p_i(n)^{(L)} \left(\sum_{k=1}^{i-1} p_k(n)^{(L)} \right)^{b-1} \, . \tag{2.12}$$

We are going to prove the beneath auxiliary lemma:

Lemma 6. *Let $n, L \geqslant 2$. Then*

$$\Pr\Big[\bar{p}(n)^{(L)}, n-1\Big] = \left(\sum_{k=1}^{L-1} p_k(n)^{(L)}\right)^{n-1} = q_L(n)^{n-1} \ .$$

Proof. Indeed, we will show even more.

Formula (2.3) holds for the distribution $\bar{p}(n)^{(L)}$, thence from (2.12) we may briefly see that for $j \in [2 : L]$ and $n \geqslant 2$, we have

$$\Pr\Big[\bar{p}(n)^{(L)}, n-1, j\Big] = \left(\sum_{k=1}^{j-1} p_k(n)^{(L)}\right)^{n-1} \ .$$

In particular, for $j = L$, we attain

$$\Pr\Big[\bar{p}(n)^{(L)}, n-1\Big] = \left(\sum_{k=1}^{L-1} p_k(n)^{(L)}\right)^{n-1} = q_L(n)^{n-1} \ ,$$

what ends the proof of Lemma 6. □

Hence from (2.11) and Lemma 6 we attain formula (2.10). The initial condition $q_1(n) \equiv 0$ can be briefly extracted from $p_1(n)^{(1)} \equiv 1$, what ends the proof of Theorem 2.5.2. □

Instantly from Eq. (2.10), for $L \geqslant 1$, we may provide the following useful formula:

$$1 - q_{L+1}(n) = \frac{1 - q_L(n)^{n-1}}{n - q_L(n)^{n-1}} \ .^{22} \tag{2.13}$$

Beneath result shows that the knowledge about the sequence $(q_i(n))_{i=1}^{L}$ entails possibility of cognition of the distribution $\bar{p}(n)^{(L)}$:

Theorem 2.5.3. *For any $n, L \in \mathbb{N}$ and $j \in [L]$, $\bar{p}(n)^{(L)}$ is given by the following formula:*

$$p_j(n)^{(L)} = (1 - q_j(n)) \cdot \prod_{i=j+1}^{L} q_i(n) \ . \tag{2.14}$$

Especially, when $j = 1$, we obtain

$$p_1(n)^{(L)} = \prod_{i=2}^{L} q_i(n) \ .$$

[22] Note that this formula is independent of the interpretation of $\Pr\big[\bar{p}(n)^{(L)}, n-1\big]$, so it can be utilized even for $n = 2$.

Proof. Note that $p_1(n)^{(1)} = 1 - q_1(n) = 1$ for $L = 1$.
Moreover, if formula (2.14) is true for some L and $j \in [L]$, then from (2.6), for $j < L$, $p_j(n)^{(L+1)} = p_j(n)^{(L)} q_{L+1}(n)$, so

$$p_j(n)^{(L+1)} = (1 - q_j(n)) \cdot \prod_{i=j+1}^{L} q_i(n) \cdot q_{L+1}(n) \ ,$$

what ends the proof. □

Note that instead of using formulas (2.4) and (2.5) to attain the optimal distribution $\bar{p}(n)^{(L)}$, we may alternatively utilize Theorem 2.5.3 together with Theorem 2.5.2.

2.5.4 Duel case ($n = 2$)

Real life solutions and problems

In real life, it is quite common to solve disputes in some form of a duel. Although, it is not a usual case of application in networks, let us consider a case $n = 2$ for a while. Remark that, in particular, such the arrangement differs from a leadership election of one amongst two candidates, which can be conducted e.g. via ballot in some specified group. We can interpret a duel case as an encounter between 2 participants of LEA.

Example 6. *Popular real life solution for this particular problem is to designate a winner by a classical "rock, paper, scissors" game. One can try to interpret this game in terms of "cheated" urn model. The "cheated" attribute is due to a randomization of a choice of a linear order of urns (labelled with symbols: rock, paper and scissors), designated to determine a winner, which have to be established via consensus between the competitors.*[23] *Although this solution is fair, we showed in Section 2.1 that such the approach is problematic when the number of competitors changes. In fact, "rock, paper, scissors" game is highly not effective when the number of player gets bigger. Even for $n = 2$, in the classic "rock, paper, scissors" with uniform and independent choice of symbols, agents tie with probability $\frac{1}{3}$. Fortunately, there exist rectified versions of this game, like e.g. "rock, paper, scissors, lizard, Spock", where there are 5 symbols and every of them beats 2 other symbols and looses to 2 other symbols. It lowers the probability of the tie to $\frac{1}{5}$, however it does not resolve efficiently the problem of many players.*

Analysis of the optimal duel in urn model

From Section 2.5.3 we already know what kind of solution should be used in repetitively fair and optimal non-anonymous LEA in urn model. First, let us provide a simple fact about $q_L(2)$:

[23] This case was mentioned in Section 2.3.2. The consensus should be emerged basing on the randomness used to draw a symbol.

Theorem 2.5.4. *For any $L \in \mathbb{N}$, there $q_L(2) = 1 - \frac{1}{L}$.*

Proof. If $L = 1$ then $1 - \frac{1}{1} = 0 = q_1(2)$. Assume that, for some $L \in \mathbb{N}$, $q_L(2) = \frac{L-1}{L}$. Then by formula (2.10), $q_{L+1}(2) = \dfrac{1}{2 - \frac{L-1}{L}} = \frac{L}{L+1} = 1 - \frac{1}{L+1}$. □

Now, we are able to provide the foregoing theorem:

Theorem 2.5.5. *For any $L \in \mathbb{N}$, $\bar{p}(2)^{(L)}$ is $\mathrm{Uni}\,(L)$ distribution.*

Proof. By the formula (2.14) and Theorem 2.5.4, we obtain

$$p_k(2)^{(L)} = (1 - q_k(2)) \cdot \prod_{i=k+1}^{L} q_i(2) = \frac{1}{k} \cdot \prod_{i=k+1}^{L} \frac{i-1}{i} = \frac{1}{L} \ ,$$

for any $k \in [L]$, so all $p_k(2)^{(L)}$ values are the same. □

A simple conclusion from Theorem 2.5.5 is that one should use uniform distribution of draws in urn model when there are only 2 competitors. Moreover, note that the probability of a tie for 2 agents is the same as for "rock, paper, scissors" ($L = 3$) and "rock, paper, scissors, lizard, Spock" ($L = 5$) games.

Let us announce that the result Theorem 2.5.5 is related to Theorem 2.6.1, where a case of anonymous arrangement of the network is considered, where an election is conducted according to $\mathrm{Uni}\,(L)$ distribution.

Let us provide a toy example of an application of Theorem 2.5.5 in a distributed procedure:

Example 7. *Assume that there are two competitors and each of them has a similar deck of* 13 *different cards with the same suit, assuming some natural linear order of cards' ranks (both of them have cards with the same number of pips or the same faces on respective cards). Each of them draws uniformly one card from self deck and sends the rank of the chosen card to the second competitor. The higher rank wins the election. Note that, when the contestants can meet, then they can draw the cards from one deck without repetitions, what eliminates the possibility of a tie. Note that in a case of non-anonymous distributed LEA for $n = 2$, the problem of eventual repetitions can be easily resolved. Namely, one of the agents should redraw (without repetition) the card which caused the tie. In a case of bigger n, the problem is slightly more complicated in distributed network arrangement.*

2.5.5 Approximation of the optimal distribution

Approximation of probabilities of non-last urns

Commence with a definition of a function $Q_n(x) = \frac{n-1}{n-x^{n-1}}$. Realize that Eq. (2.10) provides that $Q_n(q_i(n)) = q_{i+1}(n)$ for any $i \in \mathbb{N}$. Next, we show some basic properties of the sequence $(q_i(n))_{i\in\mathbb{N}}$:

Lemma 7. *The sequence $(q_i(n))_{i\in\mathbb{N}}$ is ascending and convergent to* 1.

Proof. Note that from (2.10), $q_2(n) = \frac{n-1}{n} > q_1(n) \equiv 0$.
Assume that $q_L(n) > q_{L-1}(n)$ for some $L \in \mathbb{N}\backslash\{1\}$ and calculate the following proportion:

$$\frac{q_{L+1}(n)}{q_L(n)} \overset{(2.10)}{=} \frac{n - q_{L-1}(n)^{n-1}}{n - q_L(n)^{n-1}} = 1 + \frac{q_L(n)^{n-1} - q_{L-1}(n)^{n-1}}{n - q_L(n)^{n-1}} > 1 \ .^{24}$$

Hence the sequence $(q_i(n))_{i\in\mathbb{N}}$ is rising and bounded, so convergent.
We already know that for $n, i \in \mathbb{N}\backslash\{1\}$, $q_i(n) \geqslant \frac{n-1}{n}$. Let $x, y \in [1 - \frac{1}{n}, 1]$ and $x > y$. Then

$$\begin{aligned} Q_n(x) - Q_n(y) &= (n-1)\frac{n - y^{n-1} - n + x^{n-1}}{(n - x^{n-1})\,(n - y^{n-1})} \\ &\leqslant \frac{n-1}{(n-1)^2}\left(1^{n-1} - \left(\frac{n-1}{n}\right)^{n-1}\right) \overset{n\geqslant 2}{<} \frac{1 - e^{-1}}{n-1} \ . \end{aligned}$$

Hence, for any $n \geqslant 2$, Q_n is a contraction mapping on $[1 - \frac{1}{n}, 1]$. Then, according to the Banach Fixed Point Theorem 1.4.1, $(q_i(n))_{i\in\mathbb{N}\backslash\{1\}}$ is then Banach's sequence and has a unique fixed point, which is also the limit of this sequence. We only need to note that $1 = \frac{n-1}{n-1^{n-1}} = Q_n(1)$, so from (2.10), we attain that the sequence $(q_i(n))_{i=1}^{\infty}$ converges to 1. □

From Lemma 7 we briefly see that $1 - Q_n(1) = 0$. In order to find an asymptotics of $(q_i(n))_{i=1}^{\infty}$ we may consider a behaviour of a function $1 - Q_n(1-x)$ in a vicinity of $x = 0$ to obtain an asymptotics of $(1 - q_i(n))_{i=1}^{\infty}$ instead. More precisely we are interested in finding the first coefficients of Maclaurin expansion of $1 - Q_n(1 - x)$.

Theorem 2.5.6. *For any $n \geqslant 2$*

$$q_L(n) = 1 - \frac{2}{Ln} + \frac{(2n-4)\ln(L)}{3L^2n^2} + \mathrm{o}\left(\frac{\ln(L)}{L^2}\right) \ ,$$

as $L \to \infty$.

Proof. All the calculations are made with the assumption $n \neq 1$. Then we attain:

$$-\frac{\partial Q_n(1-x)}{\partial x} = (n-1)\frac{(n-1)\,(1-x)^{n-2}}{\left(n - (1-x)^{n-1}\right)^2} \ ,$$

so $-\frac{\partial Q_n(1)}{\partial x} = \frac{(n-1)^2}{(n-1)^2} = 1$. Moreover, we get

$$\begin{aligned} -\frac{\partial^2 Q_n(1-x)}{\partial x^2} = \frac{(n-1)^2\,(1-x)^{n-3}}{\left(n - (1-x)^{n-1}\right)^3}\Big[&-(n-2)\left(n - (1-x)^{n-1}\right) \\ &-2(n-1)\,(1-x)^{n-1}\Big] \ , \end{aligned}$$

[24] Note that $n - q_L(n)^{n-1} \geqslant n - 1$, because $q_L(n)$ is a probability of not drawing L.

hence $-\frac{\partial^2 Q_n(1)}{\partial x^2} = (n-1)^2 \frac{-(n-2)(n-1)-2(n-1)}{(n-1)^3} = (-(n-2)-2) = -n$. It can be obtained that

$$-\frac{\partial^3 Q_n(1-x)}{\partial x^3} = \frac{(n-1)^2 n(1-x)^{n-4}}{\left((1-x)^{n-1}-n\right)^4}\Big[(n+1)(1-x)^{2(n-1)}+ \\ +4(n-2)n(1-x)^{n-1}+(n-3)(n-2)n\Big] \ ,$$

hence $-\frac{\partial^3 Q_n(1)}{\partial x^3} = n(n+1)$. Moreover

$$-\frac{\partial^4 Q_n(1-x)}{\partial x^4} = \frac{(n-1)^2 n(1-x)^{n-5}}{\left((1-x)^{n-1}-n\right)^5}\Big[(n+1)(n+2)(1-x)^{3(n-1)}+ \\ +(n-2)n(11n+7)(1-x)^{2(n-1)}+ \\ +(n-2)n^2(11n-29)(1-x)^{n-1}+(n-4)(n-3)(n-2)n^2\Big] \ ,$$

thence, $-\frac{\partial^4 Q_n(1)}{\partial x^4} = -n(n^2+5n-2)$ and consequently

$$1-Q_n(1-x) = x-\frac{n}{2}x^2+\frac{n(n+1)}{6}x^3-\frac{n(n^2+5n-2)}{24}x^4+\mathrm{o}\left(x^4\right) \ ,$$

as $x \to 0^+$. Now, according to S. Stević Theorem 1.4.6,

$$\begin{aligned} 1-q_L(n) &= \frac{2}{Ln}-\frac{\frac{n^2}{2}-\frac{n(n+1)}{3}}{\frac{n^2}{2}}\frac{\ln(L)}{\frac{n}{2}L^2}+\mathrm{o}\left(\frac{\ln(L)}{L^2}\right) \\ &= \frac{2}{Ln}-\frac{(2n-4)\ln(L)}{3L^2n^2}+\mathrm{o}\left(\frac{\ln(L)}{L^2}\right) \ , \end{aligned} \tag{2.15}$$

as $L \to \infty$. □

Note that in particular we may obtain

$$1-q_L(2) = \frac{1}{L}+\mathrm{o}\left(\frac{\ln(L)}{L^2}\right) \ ,$$

as $L \to \infty$, from Theorem 2.5.6, however the schema used in the proof of Stević Theorem (in [96]) may be extended in order to obtain that in fact $1-q_L(2) = \frac{1}{L}$ (what coincides with Theorem 2.5.4).

Compact form approximation of $p_j(n)^{(L)}$

It worth to mention that one can use (2.10) and (2.14) in order to provide the exact value of each $p_j(n)^{(L)}$. However, it occurs, that usually, i.e. when n or L is big, such the calculation lasts very long and can be burdened with numerical errors. According to (2.15), we may assume that

$$q_L(n) \approx 1-\frac{2}{nL} \ .$$

Hence, from (2.14), for $j \in [L]$, we attain

$$p_j(n)^{(L)} = (1 - q_j(n)) \cdot \prod_{i=j+1}^{L} q_i(n) \approx \frac{2}{nj} \cdot \prod_{i=j+1}^{L} \left(1 - \frac{2}{ni}\right) .$$

Then we can calculate (see Section 1.2.6 and Section 1.2.9 for approximations of logarithms and harmonic numbers)

$$\begin{aligned} \ln\left(p_j(n)^{(L)}\right) &\approx \ln\left(\frac{2}{nj}\right) + \sum_{i=j+1}^{L} \ln\left(1 - \frac{2}{ni}\right) \approx \ln\left(\frac{2}{nj}\right) - \frac{2}{n} \sum_{i=j+1}^{L} \frac{1}{i} \\ &= \ln\left(\frac{2}{nj}\right) - \frac{2(H_L - H_j)}{n} \approx \ln\left(\frac{2}{nj}\right) - \frac{2}{n} \ln\left(\frac{L}{j}\right) , \end{aligned}$$

so for $j \in [L]$, we may provide

$$p_j(n)^{(L)} \approx \frac{2}{nj} \left(\frac{j}{L}\right)^{\frac{2}{n}} . \tag{2.16}$$

The approximation (2.16) should be more accurate for bigger values of j, because it arises from the asymptotical formulas as $L \to \infty$. Note that for $n = 2$, (2.16) gives exactly the demanded distribution $p_j(n)^{(L)} = \frac{1}{L}$.

Establishment of errors of the proposed approximation is postponed to future work.

In Appendix A, we will provide some examples of numerical representations of optimal distributions and efficient approximations of optimal solutions, which can be obtained much faster without a significant loss of probability of success.

2.5.6 A number of bits for NALEA according to the optimal distribution

Let us search for an optimal solution for a given number of bits $K := \lceil \lg L \rceil$, in order to find minimal necessary memory. Let us consider $(1 - \varepsilon)$-reliable LEA, with an assumption that a number of devices is exactly n. The definition of $\bar{p}(n)^{(L)}$ and Lemma 6 show that $q_L(n)^{n-1} = \Pr\left[\bar{p}(n)^{(L)}, n-1\right] \leqslant \Pr\left[\bar{p}(n)^{(L)}, n\right]$. We will show that $q_L(n)^{n-1}$ is non-increasing with respect to n.

Lemma 8. *For $0 \leqslant x \leqslant 1$ a sequence $\left(\left(\frac{k}{k+x}\right)^k\right)_k$ is non-increasing.*

Proof.

$$\frac{\left(\frac{k}{k+x}\right)^k}{\left(\frac{k+1}{k+1+x}\right)^{k+1}} = \left(\frac{k(k+1+x)}{(k+x)(k+1)}\right)^k \frac{k+1+x}{k+1} = \dots$$

$$\begin{aligned}\ldots &= \left(\frac{k^2+kx+k}{k^2+kx+k+x}\right)^k \frac{k+1+x}{k+1} \\ &= \left(1-\frac{x}{(k+x)(k+1)}\right)^k \frac{k+1+x}{k+1}\ .\end{aligned}$$

Note that the expression in parentheses is from the interval $(0,1)$, so from Bernoulli Inequality (Theorem 1.4.7), we can obtain $\left(1-\frac{x}{(k+x)(k+1)}\right)^k \geqslant 1-\frac{xk}{(k+x)(k+1)}$. Therefore

$$\begin{aligned}\frac{\left(\frac{k}{k+x}\right)^k}{\left(\frac{k+1}{k+1+x}\right)^{k+1}} &\geqslant \left(1-\frac{kx}{(k+x)(k+1)}\right)\left(1+\frac{x}{k+1}\right) \\ &= 1+\frac{x^2}{(x+k)(k+1)}\left(1-\frac{k^2+kx}{k^2+kx+k+x}\right) \\ &= 1+\frac{x^2}{(x+k)(k+1)^2} \geqslant 1\ .\end{aligned}$$

□

Theorem 2.5.7. *A sequence* $\left(q_L(n)^{n-1}\right)_{n\in\mathbb{N}}$ *is non-increasing.*

Proof. For $L=1$, $q_L(n)=0$, independent on n. Assume that for some L, $q_L(n)^{n-1} \geqslant q_L(n+1)^n$. Then, according to Eq. (2.10), $q_{L+1}(n+1)^n = \left(\frac{n}{n+1-q_L(n+1)^n}\right)^n \leqslant \left(\frac{n}{n+1-q_L(n)^{n-1}}\right)^n$. From Lemma 8, for $k=n-1$ and $x=1-q_L(n)^{n-1}$ we have

$$\frac{q_{L+1}(n)^{n-1}}{q_{L+1}(n+1)^n} \geqslant \frac{\left(\frac{n-1}{n-q_L(n)^{n-1}}\right)^{n-1}}{\left(\frac{n}{n+1-q_L(n)^{n-1}}\right)^n} \geqslant 1\ ,$$

what ends the proof. □

Note that Theorem 2.5.7 together with Lemma 6 and Theorem 2.5.4 entail that $q_L(n)^{n-1} = \Pr\left[\bar{p}(n)^{(L)}, n-1\right] \leqslant \Pr\left[\bar{p}(n)^{(L)}, n\right] \leqslant 1-\frac{1}{L}$, so the minimal number of bits K per agent needed to perform $(1-\varepsilon)$-reliable NALEA is $\lceil -\lg(\varepsilon)\rceil$. Moreover, from Lemma 7, $q_L(n)$ is the increasing sequence (with respect to L), so from Theorem 2.5.6, we can obtain $\Pr\left[\bar{p}(n)^{(L)}, n\right] \geqslant q_L(n)^{n-1} \geqslant \left(1-\frac{2}{Ln}\right)^{n-1} > 1-\frac{2}{L}$,[25] in what shows that $K = \lceil -\lg(\varepsilon)\rceil + 1$ bits are enough to perform any optimal non-anonymous LEA.

[25] In fact, one can use the technique presented in [96] in order to expand $\mathrm{o}\left(\frac{\ln L}{L^2}\right)$ part from Theorem 2.5.6 to a form of $\sum_{j=2}^{\infty} \frac{c_j}{L^j}$, where all c_j are positive constants which can be described in a language of Riemann's Zeta function (see Section 1.2.10 for definition) and generalized harmonic numbers ${H_n}^{(k)}$, where $k \in \mathbb{N}$ (see Section 1.2.9 for definitions). The proof of this

Example 8. *Consider a simple example, where* $n = 3$ *and* $\varepsilon = \frac{1}{2}$. *We will consider two cases:* $L = 2$ $(K = 1)$ *and* $L = 4$ $(K = 2)$. *According to (2.10), one can see, that* $\bar{p}(3)^{(2)} = \left(\frac{2}{3}, \frac{1}{3}\right)$ *and* $\bar{p}(3)^{(4)} = \left(\frac{184}{421}, \frac{92}{421}, \frac{230}{1263}, \frac{205}{1263}\right)$ *and, by formula (2.1), the probabilities of successful Leader Election are respectively* $\frac{4}{9}$ *and* $\frac{1119364}{1595169} = 0.70172\ldots$. *Therefore one bit* $(\lceil \lg(\varepsilon^{-1}) \rceil = 1)$ *is not enough to perform* $\frac{1}{2}$*-reliable NALEA for* $n = 3$, *however* $K = 2$ *is sufficient. This shows that sometimes* $\lceil \lg(\varepsilon^{-1}) \rceil + 1$ *bits are necessary.*
Nevertheless, note that, when $\varepsilon = 0.49$ *instead of* 0.5, *then* $\lceil \lg(\varepsilon^{-1}) \rceil = 2$, *so for* $n = 3$, *the lower bound is realized.*

2.6 Uniform Leader Election (ULE)

2.6.1 Basic properties

In Section 2.5 we have obtained some optimal and approximated results for LEA when the number of agents n is known by all the nodes in the network. We have also obtained that in non-anonymous, urn model, the tossing generator with uniform distribution is the best possible one when there are exactly 2 competitors (Theorem 2.5.5). Moreover in this arrangement, from Theorem 2.5.3 and Lemma 7 we can presume that for "big supports" $[L]$ of the optimal distribution, $p_j(n)^{(L)} \approx p_{j-1}(n)^{(L)}$, as j is "relatively close" to L[26]. This conjecture together with Theorem 2.5.5 expound our next goal. Namely, we would like to consider a quasi-anonymous LEA in urn model, where the tossing generator follows $D(\Theta) = \mathrm{Uni}\,(L)$ distribution (according to notation from Algorithm 2).[27] Therefore we call this solution Uniform Leader Election (or shortly ULE).

We commence our considerations from the reformulation of Theorem 2.4.1:

Theorem 2.6.1. *Let* $n \in \mathbb{N}\backslash\{1\}$ *and* $X_1, \ldots, X_n$ *be a sequence of i.i.d. random variables, where* $X_1 \sim \mathrm{Uni}\,(L)$. *Let* $S_{n,\mathrm{Uni}(L)}$ *denotes the successful ULE algorithm for* n *devices, i.e. the event* $(\exists\, k)(\forall\, j \neq k)\ (X_j < X_k)$. *Then*

$$\Pr\left[S_{n,\mathrm{Uni}(L)}\right] = \frac{n}{L^n} \sum_{j=1}^{L-1} j^{n-1}\ . \tag{2.17}$$

Note that the result analogous to (2.17) was claimed in [43], for slightly different LEA, based on the idea from [89]. More details will be provided in Section 2.7.1.

---

fact is a recursive application of the proof of Stević Theorem 1.4.6, hence even the exact formula for c_2 is quite difficult to obtain, but one can relatively easy verify, that all elements c_j (for $j \in \mathbb{N}\backslash\{1\}$) are positive. Since the exact form of the constants is not very informative, we abandon this purely technical proof.

[26] One can justify this approximation, basing on tables of distributions from Appendix A or the universality and monotonicity of sequences $(a_i(n))_{i\in\mathbb{N}}$ from Section 2.5.2

[27] We may alternatively utilize $\mathrm{Uni}\,(A)$, where A is some linearly ordered set of cardinality L.

Proof. First of all, notice that for any $i \in [n]$, $\Pr[X_i = j] = \frac{1}{L}$ and $\Pr[X_i < j] = \frac{j-1}{L}$. Hence from Theorem 2.4.1 we get

$$\Pr\big[S_{n,\mathrm{Uni}(L)}\big] = n\sum_{j=2}^{L}\frac{1}{L}\left(\frac{j-1}{L}\right)^{n-1} = \frac{n}{L^n}\sum_{j=1}^{L-1} j^{n-1}\ .$$

□

Theorem 2.6.2. *With the same notations and assumptions as in Theorem 2.6.1 we obtain*

$$\Pr\big[S_{n,\mathrm{Uni}(L)}\big] = \sum_{j=0}^{n-1}\binom{n}{j}\frac{B_j}{L^j} \ = 1 - \frac{n}{2L} + R_2\ ,$$

where $0 \leqslant R_2 \leqslant \frac{1}{6}\left(\frac{n}{L}\right)^2$.

Proof. The sum $\sum\limits_{j=1}^{L-1} j^{n-1}$ can be expressed by the classical Faulhaber's formula (see Eq. (1.13)):

$$\sum_{j=1}^{L} j^{n-1} - L^{n-1} = \frac{L^n}{n} - \frac{L^{n-1}}{2} + \sum_{j=2}^{n-1}\binom{n}{j}\frac{B_j L^{n-j}}{n} = \frac{L^n}{n}\sum_{j=0}^{n-1}\binom{n}{j}\frac{B_j}{L^j}\ .$$

However, computationally it begins to be unwieldy as n becomes large. Instead of coping with this approach, we use Euler—Maclaurin summation formula (1.14) for the polynomial $w(x) = x^{n-1}$:

$$\sum_{j=0}^{L-1} j^{n-1} = \sum_{j=0}^{L} j^{n-1} - L^{n-1} = \frac{1}{n}L^n - \frac{1}{2}L^{n-1} + \frac{n-1}{12}L^{n-2} + r_2\ , \qquad (2.18)$$

where $r_2 = -\int\limits_0^L \frac{B_2(\{x\}_f)}{2}(n-1)(n-2)x^{n-3}dx$ (by (1.15)) and $B_2(t) = t^2 - t + \frac{1}{6}$ is the second Bernoulli polynomial. By inequality (1.12) and Fact 1.2.13, we get $|B_2(t)| \leqslant \frac{1}{6}$ for $t \in [0,1]$, so $|r_2| \leqslant \frac{n-1}{12}L^{n-2}$. The final equation follows directly from this upper bound and Theorem 2.6.1.[28] □

2.6.2 Monotonicity of success probability

It is intuitive that the chance of becoming a leader should be lower, when there are more competitors, that is $\Pr\big[S_{n,\mathrm{Uni}(L)}\big]$ should decrease with respect to n. It is formally confirmed by the beneath:

[28] One can simply rectify the result to a more precise one. Since the odd elements of Bernoulli's sequence are zeros (except B_1), therefore we may simply obtain an alternative bound via the same technique, where r_2 is substituted by r_3, and by (1.12) and Fact 1.2.13, one can get $|r_3| < \frac{(n-1)(n-2)}{100}L^{n-3}$. Then $\Pr\big[S_{n,\mathrm{Uni}(L)}\big] = 1 - \frac{n}{2L} + \frac{n(n-1)}{12L^2} + R_3$ with $|R_3| < \frac{1}{100}\left(\frac{n}{L}\right)^3$.

Theorem 2.6.3 (Monotonicity)**.** *With the same notations as in Theorem 2.6.1, the following inequality holds* $\Pr\left[S_{n,\mathrm{Uni}(L)}\right] > \Pr\left[S_{n+1,\mathrm{Uni}(L)}\right]$ *for each* $n \geqslant 2$ *and* $L \geqslant 2$.

Proof. Let us introduce an abbreviation:

$$\Delta_{n,L} := \sum_{j=1}^{L-1} \left((n+1)j^n - Lnj^{n-1}\right) \ .$$

Observe that $\Pr\left[S_{n+1,\mathrm{Uni}(L)}\right] - \Pr\left[S_{n,\mathrm{Uni}(L)}\right] = \frac{\Delta_{n,L}}{L^{n+1}}$. Therefore, our goal is to show that $\Delta_{n,L} < 0$. At start, realize that $\Delta_{n,2} = 1 - n < 0$. Let us define another auxiliary notation — $D_{n,L} := \Delta_{n,L+1} - \Delta_{n,L}$. Fix n and note that, if for each $L \geqslant 2$, $D_{n,L} < 0$, then Theorem 2.6.3 will be proved.

From the definition, we attain

$$\begin{aligned} D_{n,L} &= \sum_{j=1}^{L} \left((n+1)j^n - (L+1)nj^{n-1}\right) - \sum_{j=1}^{L-1} \left((n+1)j^n - Lnj^{n-1}\right) \\ &= (n+1)L^n - (L+1)nL^{n-1} - \sum_{j=1}^{L-1} nj^{n-1} = L^n - n\sum_{j=1}^{L} j^{n-1} \ . \end{aligned}$$

From (1.3) we obtain $\sum\limits_{j=1}^{L} j^{n-1} > \int\limits_{0}^{L} x^{n-1}dx = \frac{1}{n}L^n$ (alternatively from (2.18)), hence $D_{n,L} < L^n - n\frac{L^n}{n} = 0$, what ends the proof. □

It worth to mention that, to the best of our knowledge, this type of monotonicity result was not considered before, although, it seems quite intuitive.[29] However, not every LEA has this property (see e.g. Section 2.7.3 or Section B.4). This problem seems rather easy at the first glance, however, to highlight the level of difficulty, let us note, that this is on of two of considered LEAs, for which we were able to provide such the property (the other one is provided in Section 2.9).

2.6.3 Miscellaneous remarks and main result

Remark 1 We should use Theorem 2.6.2 only when $n \ll L$. Intuitively, in this case the approximation $\Pr\left[S_{n,\mathrm{Uni}(L)}\right] \approx 1 - \frac{n}{2L}$ is very precise. Figure 2.3 shows the result of numerical experiments with $n = 5$ and L varying from 1 to 500. The grey line is the plot of the function $y(L) = 1 - \frac{5}{2L}$.

[29] Often, the authors provide some bounds on the probability of success (usually they utilize Landau notation), which are monotonic with respect to the number of competitors. However, dispute on the monotonicity property for the probabilities of success is usually omitted.

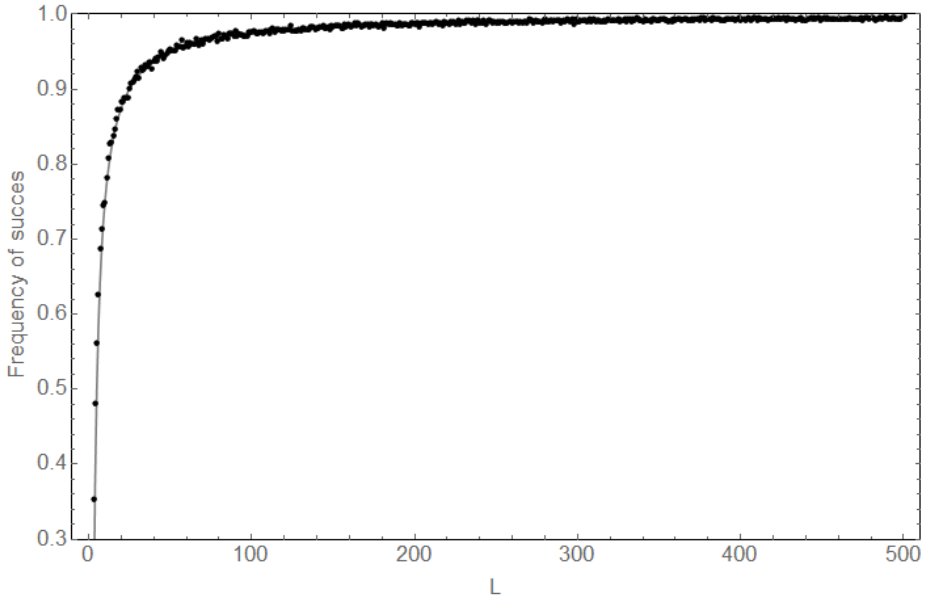

Figure 2.3: Precision of the approximation $\Pr\left[S_{n,\mathrm{Uni}(L)}\right] \approx 1 - \frac{n}{2L}$ for $n = 5$ and L varying from 1 to 500. For each L we carried out 5000 experiments and calculated the frequency of success.

Remark 2 Let us compare our result with the probability $b_{n,L}$ that a random uniform sample, which consists of n elements from the universe of L elements, has no duplicates (note that this is a general arrangement of the Birthday Paradox). Namely, for $n \ll \sqrt{L}$, we have $b_{n,L} \approx 1 - \frac{n^2}{2L}$ (see e.g [47]). The goal of reliable LEA is to omit duplicates only for the biggest chosen element, so naturally $b_{n,L} < \Pr\left[S_{n,\mathrm{Uni}(L)}\right] \approx 1 - \frac{n}{2L}$.

Remark 3 Consider a connected graph with countable number of nodes. Let us think about a process, in which nodes choose theirs labels according to $\mathrm{Uni}\,(L)$ distribution, one by one, until some node draws the same value as a previous maximum. Then the process stops. Consider an expected stopping time $\mathbb{E}[C_L]$ of such the process. A first vertex establishes the first maximum. Every next node has the probability $\frac{1}{L}$ to draw the present maximum. That implies $\mathbb{E}[C_L] = 1 + \mathbb{E}\left[\mathrm{Geo}\left(\frac{1}{L}\right)\right] = L + 1$. Hence, for $L \gg 1$, we should rather expect the first collision after about L steps.
However, in practise, the network has finite cardinality n and if $n \ll L$ (see Remark 1), then a duplicate rather should not happen. Remark that, when a repetition occurs after $k < n$ steps and we continue the drawing process until we reach n steps, then the new unique maximum may be still find after the step k, which resolve the problem of Leader Election in such the arrangement

although the repetition occurred before. The smaller is the present maximum at k-th time step, the bigger is the chance that it will be resolved until the n-th round.

Remark 4 Let us finally consider $(1-\varepsilon)$-reliability of ULE. As we have seen in previous Remark, when one increase the value of L, the probability of collision lowers, so let $L = 2^K$ in order to optimize the support for a given number of rounds K. Notice that according to Theorem 2.6.2, we would like to fulfil $\Pr\left[S_{n,\mathrm{Uni}(L)}\right] \geqslant 1 - \frac{n}{2^{K+1}} \geqslant 1 - \varepsilon$, where the second inequality is equivalent to

$$K \geqslant \lg(n) + \lg\left(\varepsilon^{-1}\right) - 1 \ . \tag{2.19}$$

This way we have obtained a lower bound on the number of bits of memory necessary for the execution of $(1-\varepsilon)$-reliable ULE exactly for n agents. Note that this result coincides with the one from Section 2.5.4, i.e. when $n = 2$, then $K \geqslant \left\lceil \lg\left(\varepsilon^{-1}\right)\right\rceil$. Nevertheless, in general, there is approximately $\lg(n) - 1$ overhead relative to the optimal solutions for non-anonymous LEA (see Section 2.5.6). However we have assumed quasi-anonymity of the network, so let N be the capacity of the network known a priori by all the nodes. Then, from Theorem 2.6.3, K should be at least $\lg(N) + \lg\left(\varepsilon^{-1}\right) - 1$ (it should be also intuitive from the inequality (2.19)). Similarly as in Section 2.5.6, a role of factor $\lg\left(\varepsilon^{-1}\right)$ seems inherent and it takes an essential share in K (especially when we want ε to be very small), side by side with the factor $\lg(N)$.

Formulation of main result for ULE

Now we are ready to formulate a main theorem of this section, which provides the parameters for quasi-anonymous ULE algorithm:

Theorem 2.6.4. *Let $0 < \varepsilon < 1$ and $N \in \mathbb{N}$. Moreover, let*

1. $K = \left\lceil \lg(N) + \lg\left(\varepsilon^{-1}\right)\right\rceil - 1$,
2. $L = 2^K$.

Then for every $n \in [N]$, the following is satisfied $\Pr\left[S_{n,\mathrm{Uni}(L)}\right] \geqslant 1 - \varepsilon$.

From Theorem 2.6.4, we can easily obtain the number of bits needed by the devices in order provide $(1-\varepsilon)$-reliable quasi-anonymous ULE algorithm.

2.6.4 Role of the factor $\lg\left(\varepsilon^{-1}\right)$

We have highlighted that the factor $\lg(\varepsilon^{-1})$ plays a crucial role in the estimation of a proper parameter K of ULE. Beneath we are going to explicate this thread in general.

Theorem 2.6.5. *Let $0 < \varepsilon < 1$ and $N \in \mathbb{N}$. Let D be a probability distribution with linearly ordered support of cardinality L and $S_{n,D}$ be an event of successful LEA in urn model, where tossing generators draw identities independently*

according to distribution D. If $(\forall\, n \leqslant N)\ \Pr[S_{n,D}] \geqslant 1-\varepsilon$, then the minimal number of bits K that are needed to save any identity drawn according to distribution D is given by the formula $\lceil \lg(\varepsilon^{-1}) \rceil$.
Moreover K is non-decreasing with respect to N.

Proof. Consider a case when $n = 2$ nodes generates an i.i.d. realization of the same random variable ranged in $[2^K]$ and LEA in urn model decides which of the nodes become a leader. This procedure will fail if both of them choose the same numbers. We can model this process as follows: there are two independent random variables X and Y with the same distribution ranged in $[2^K]$. Then

$$\Pr[X = Y] = \sum_{i \in [2^K]} p_i^2 \ .$$

Observe that from Cauchy—Schwarz inequality (Theorem 1.4.4), we obtain

$$1 = \left(\sum_{i \in [2^K]} p_i \right)^2 \leqslant \left(\sum_{i \in [2^K]} p_i^2 \right) \left(\sum_{i \in [2^K]} 1^2 \right) = 2^K \Pr[X = Y] \ ,$$

so $\Pr[X = Y] \geqslant 2^{-K}$. Therefore, a constraint $\Pr[X \neq Y] \geqslant 1 - \varepsilon$ implies that one needs $K \geqslant \lg(\varepsilon^{-1})$ bits of memory in order to break a tie with probability at least $1 - \varepsilon$ in urn model. However, note, that in quasi-anonymous arrangement, when one increase the network's capacity N, then one get additional constraints. In consequence, an optimal probability distribution (for a given N), which maximizes $\min\{\Pr[S_{n,D}] : n \leqslant N\}$, may only enlarge the efficiency rate, when one wants to guarantee $(1-\varepsilon)$-reliability of the quasi-anonymous LEA (if the solution for smaller N is optimal, then it has minimal number of bits, so after the increase of N, the number of bits cannot be smaller). The above argument shows that $K \geqslant \lg(\varepsilon^{-1})$ for any quasi-anonymous LEA. □

Remark 5 Realize that when $n = 2$, then $\lg(\varepsilon^{-1}) = \lg(n) + \lg(\varepsilon^{-1}) - 1$ which is the number of rounds K of ULE algorithm. According to Section 2.5.4, $\Pr\left[S_{2,\mathrm{Uni}(2^K)}\right] = 1 - \frac{1}{2^K}$. One may easily see that this result coincides with the approximation provided for ULE in a case of $n = 2$ in Theorem 2.6.2.

This shows that the lower bound provided in Theorem 2.6.5 is reachable and that in Theorem 2.6.2, R_n may be 0 (the worst possible case).

Remark 6 Realize that, according to Theorem 2.6.4, a node which becomes a leader has to send at most $K \approx \lg\left(\frac{N}{2\varepsilon}\right)$ BEEPs. However, when we think of total energy cost (number of all BEEPs), then it occurs to be $\mathrm{O}(N)$, as $N \to \infty$. Intuitively, in first round half of the devices sends BEEP on average, because there is the same probability that k and $n-k$ devices transmit. Due to the same argument, a quarter part of all devices send a signal on average (if k of devices were transmitting in the first round, then $\frac{k}{2}$ transmit in the second one on average and with the same probability $\frac{n-k}{2}$ transmit in this round)[30], and

[30]The only exception is when all devices in the first round are listening to, however, this event has negligible probability.

so on. This argument gives $\sum_{i=1}^{K} \frac{n}{2^i} < N$ BEEPs plus some negligible number of additional signals, which are a consequence of the rounds, in which everybody are listening to.

2.7 Previous Developments of Leader Election

2.7.1 How to select a loser?

Description of Prodinger's algorithm

One of the first remarkable works that can be assigned to a class of universal randomized LEAs is the classical Prodinger's algorithm from [89], written in 1993. In here, we would like to provide a brief description of this approach.

At start of the procedure, there is a party of n contestants. Everyone is tossing a fair coin,[31] where "heads" are denoted by '1' and "tails" are denoted by '0'. After this step, there are two groups obtained: '0'-party and '1'-party (each agent is present in the party marked by his last toss). The procedure is then repeated within the '0'-party if it is not empty or in '1'-party otherwise. The last man standing in the '0'-party becomes the loser. It is an example of oblivious Las Vegas algorithm. It was shown that on average, $\lg(n)+\frac{1}{2}+\delta(\lg(n))$ tossing rounds are needed in order to provide the loser, where δ is some periodic function with very small amplitude.[32] In the nomenclature of beeping model, that means, that we need $K \approx \lg(n) + \frac{1}{2}$ bits on average in order to terminate the election (of a loser) successfully. Moreover, Prodinger showed that there are $2n$ coin tosses on average in his model. Hence the expected total energy cost is exactly n.

First of all, note that if we substitute the term "loser", with "leader", we will obtain a randomized LEA. Moreover, we can artificially assume that when at some round an agent tosses '1' and '0'-party is not empty, then this agent will continue tossing, but it does not count into the election process. This way every device generates a sequence of tosses or alternatively '0's and '1's and when the procedure stops, each of devices has binary sequence of the same length K. When we swap '1's with '0's and write down such the sequences from left to right, then each agent will have a binary representation of length K of some number form the interval $[0 : 2^K - 1]$. Realize that this solution can be also investigated in terms of Algorithm 2, where $L = 2^K$ and $D(\Theta) = \mathrm{Uni}\,(L)$. Note that in beeping model, when a device hears a BEEP or collision at some round, then he resigns from being the leader and stops the transmission. A similar situation occurs in Prodinger's algorithm — if an agent has heads at some round and sees that there are other contestants, who have tails, then he resigns from further tossing. Since the function BIN_K is a bijection between $\{0,1\}^K$ and the set $[2^K]$, we deduce that the binary sequences which control the

[31] Both probability of heads and tails equals to $\frac{1}{2}$.

[32] For more details see [89].

behaviour of the nodes are generated uniformly (and independently). However, note that K is unknown in advance and is unbounded.

Efficiency of restricted version of Prodinger's algorithm

Note that we may obtain a restricted version of Prodinger's algorithm, where we resolve the aforementioned problem by proceeding exactly K tossing rounds for some a priori chosen parameter K. The previous argument bears that such the restricted version of algorithm is equivalent with ULE.[33]

We may apply Theorem 2.6.1 to deduce that the probability of successful election of a leader by restricted Prodinger's algorithm is given by formula

$$\Pr\left[S_{n,\mathrm{Uni}(2^K)}\right] = \frac{n}{2^K} \sum_{j=1}^{2^K-1} j^{n-1} \approx 1 - \frac{n}{2^{K+1}} \ ,$$

where the last approximation is very precise when $n \ll 2^K$.

Moreover, the equivalence of restricted version of Prodinger's algorithm and ULE one justifies the Remark 6 from Section 2.6.4, that the expected total energy cost of ULE algorithm is at most N.

2.7.2 Leader Green Election

In Section 2.6 we have considered LEA, where identities of the devices were drawn according to $\mathrm{Uni}\,(L)$ distribution in urn model.

Leader Green Election (LGE) was introduced by P. Jacquet et al. in [64]. Here we present an idea of the **main block** of this algorithm: fix a small number $p \in (0,1)$ (say, as authors propose, $p = 10^{-2}$), use Algorithm 2 with $D(\Theta) = \mathrm{Geo}\,(p)$ and $L = \infty$. LGE is Las Vegas algorithm, however, when one wants to restrict it in order to obtain Atlantic City version of LGE, then one need to choose some sufficiently large parameter L.

When one use $\mathrm{Geo}\,(p)$ distribution in urn model of LEA, then the probability of a conflict is circa $\frac{p}{2}$ (see Theorem 1.3.3), hence authors of [64] advise to use the main block several times in order to reduce the mistake frequency. More precisely, they suggest to repeat it k times, where k is such that $\left(\frac{p}{2}\right)^k$ is sufficiently small (e.g. $< \varepsilon$ according to our definition of mistake frequency from Section 2.1). Besides, some confusion is connected with the choice of parameter L and will be dispelled later.

Let us announce that in Section 2.8 we will provide more efficient, restricted solution, which execute a single run with appropriately chosen parameter p of the applied geometric variables. This changes sounds insignificantly at first, however a precise analysis entails somewhat astonishing effect. In particular, we will establish p depending on a maximal size of the network N (given a priori)

[33] There is a subtle difference — from a technical point of view, restricted Prodinger's algorithm is non-anonymous and ULE is quasi-anonymous. However, both can be considered in both arrangements without a loss of quality.

and the required reliability. Usually the parameter p occurs to be much smaller than 10^{-2} (proposed by Jacquet). Moreover, contrary to LGE approach, we run the procedure only once and know the number of transmission rounds K in advance. Realize that, as we mainly focus on the number of bits needed to save the identities, the repetitions of the procedure (like in LGE algorithm) may force us to conduct many unwanted rounds when we want to reach the required accuracy.

Propositions of rectifications of LGE algorithm

In this part, we are going to dilate the discussion from [21], since our further contribution is a natural continuation of this paper. For attention, let us fix $p = 0.01$, network's capacity $N = 10^{26}$ and mistake frequency $\varepsilon = 10^{-23}$.

Notice that one can employ Fact 1.3.7 in order to conclude that if $M \sim \mathrm{MGeo}(N, p)$ then $\Pr\left[M > \frac{2\ln N}{\ln\left(\frac{1}{1-p}\right)}\right] < N^{-1}$.[34] When we put our parameters into this formula, then we get $\Pr[M > 12500] < 10^{-26}$, what is negligible from a practical point of view. Therefore we expect that during the execution of LGE algorithm for N contestants, there will not be any realization of geometric distribution above 12500. Then $K = \lfloor \lg(12500) \rfloor + 1 = 14$, hence only 14 bits are required to run the main block of LGE algorithm in the framework of Algorithm 2 with $D(\Theta) = \mathrm{Geo}(0.01)$ to achieve the reduction with probability of at least $1 - 10^{-26}$.[35] Note that the less devices participate in LGE, the lower is $\Pr[M > 12500]$. Hence one can choose $L = 12500$ by an additional cost of less than 10^{-26} included in the mistake frequency.
Let $S_{n,\mathrm{Geo}(0.01,12500)}$ denotes an event of successful LEA according to the main block of LGE algorithm. In terms of Section 1.3.10, this event is equivalent to $W_{1,n,p}$.[36] Theorem 1.3.3 shows that $\Pr\left[S_{n,\mathrm{Geo}(0.01,12500)}\right] \approx 1 - \frac{0.01}{2}$, hence the probability of failure is quite large. This is the reason why authors of [64] suggested to use the main block of procedure 10 times for $p = 0.01$. Then the probability of success of restricted version[37] of LGE should be at least $1 - \left(\frac{0.01}{2}\right)^{10} - 10 \cdot 10^{-26} > 1 - 10^{-23}$. This way we have provided a restricted, quasi-anonymous, Atlantic City version of Leader Green Election, which is $\left(1 - 10^{-23}\right)$-reliable and uses $K = 10 \cdot 14 = 140$ bits of memory per device. When one use $L = 2^{14} - 1$ instead of 12500, then the algorithm will be more reliable and the number of bits remains the same. It worth to mention that this solution is fully-anonymous Atlantic City algorithm.

One can easily generalize the above approach via manipulations of parameters p, N, ε and the number of repetitions. However, a more difficult task is to optimize the solution with respect to those parameters. We will explicate

[34] Note, that one can tune up the constant C from Fact 1.3.7 in order to obtain other bounds, respective to N assumed a priori.

[35] Since the support of $\mathrm{Geo}(p)$ is $\mathbb{N}_0$, therefore, the restriction by the operation $\min(., L)$ entails that $K = \lceil L + 1 \rceil$.

[36] See Section 1.3.10 for definition of the event

[37] Every main block is restricted to [12500].

the above idea partially in Section 2.8, where we use a single run of a carefully modified LGE.

We can still use the main block of LGE in more efficient way. Namely, from Theorem 1.3.4, we may deduce that

$$\Pr\left[W_{n,\frac{1}{100}} \geqslant 13\right] < 7.823 \cdot 10^{-26}$$

for any $n > 1$. Thence we may treat main block of LGE as a method for a quick reduction of an arbitrary collection of nodes to a small subgroup (when $p = 0.01$, then with probability at least $1 - 7.823 \cdot 10^{-26}$ this subgroup has cardinality at most 12). This evince that the most difficult part of LGE is to determine the unique maximum for small values of n. To develop a really efficient algorithm, we have to deal properly with quasi-anonymous Leader Election for small N (in this case: $N = 12$). Namely, we want Leader Election to be juxtaposed two phases: the first one is restricted main block of LGE algorithm according to $\mathrm{Geo}\,(p, L)$ distribution (with e.g. $L = 2^{14} - 1$, as proposed earlier) and the another one (which is quasi-anonymous LEA per se) with some other distribution, which resolves the problem efficiently for small N.

Authors of [21] suggested to use solutions provided in [89] (fair coin toss described in Section 2.7.1), in [65, 76] (biased coin toss). However they did not investigated these solutions. Let us slightly dilate into some of the cases.

Since restricted version of solution from [89] is equivalent with ULE algorithm, let us commence with this case. In Section 2.6, we have showed that ULE algorithm is quite efficient for $N \ll L$, so especially for small N. We have to remember about the first phase of Leader Election, in which we utilize $\mathrm{Geo}\,(p, L)$ distribution, which deteriorates the admissible mistake frequency (for assumed parameters, the reduction is at most $7.823 \cdot 10^{-26} + 10^{-26}$)[38]. We naturally have to care the reduction to be less than ε assumed a priori. Let us denote the difference between ε and the reduction by $\tilde{\varepsilon}$. Recall that in ULE phase, the number of needed bits is $K = \lceil \lg(N) - \lg(\tilde{\varepsilon}) \rceil - 1$. Also note that $\lg(12) - 1 = 2.58496\ldots$, so this solution will give at most $K = \lceil -\lg(\tilde{\varepsilon}) \rceil + 3$ bits.[39] In our case we have

$$\lceil -\lg(\varepsilon) + \lg(12) \rceil - 1 = \lceil 76.4043\ldots + 3.5849\ldots \rceil - 1 = \lceil 78.9893\ldots \rceil = 79$$

and

$$\lceil -\lg(\tilde{\varepsilon}) + \lg(12) \rceil - 1 = \lceil 76.4171\ldots + 3.5849\ldots \rceil - 1 = \lceil 79.0021\ldots \rceil = 80\ .$$

This shows that the in some situations, we may lose an additional bit of memory. We can eventually modify N parameter of ULE phase or L in the $\mathrm{Geo}\,(p, L)$, but it may eventually increase the number of needed bits even more, so it has to be carried out carefully.

The solution presented above uses $14+80 = 94$ bits of memory per device, which

[38] We have to add the probability 10^{-26} that the maximum exceeds L

[39] When one cares to keep $\varepsilon \approx \tilde{\varepsilon}$, then $\lceil -\lg(\tilde{\varepsilon}) \rceil = \lceil -\lg(\varepsilon) \rceil$.

effectively reduces the previous one (140). We highlight that this time we also deal with fully-anonymous oblivious Atlantic City LEA. The above example is a presage of an algorithm, which we will consider in Section 2.9.

Let us note, that if the first phase does not return a single winner, then it is the most probable (circa $\frac{p}{2} = 0.005$) that there will be two competitors in the second phase. Section 2.5.4 have showed that in this case the best solution is ULE algorithm. The probability that there will be at least 3 contestants in the second phase is given by Theorem 1.3.4: $\Pr[W_{n,0.01} \geqslant 3] = 3.38478\ldots \cdot 10^{-5}$. This justifies the approach that the distribution utilized in the second phase should be efficient mainly for the case $n = 2$.

In the second phase of efficient algorithm one can alternatively try to use the optimal solutions from Section 2.5 like e.g. $\bar{p}(12)^{(L)}$ for some L or theirs approximations (provided in Section A.2). Another idea is to use a convex combination of such the distributions. However we postpone the analysis of the above ideas to future work.

2.7.3 Distributed splitting and naming procedures

Now we are going to analyze one of the most recent solutions, from 2015. Consider a fully-anonymous network with n devices. In [79], Métivier, Robson and Zemmari proposed two, closely similar Las Vegas LEAs that fit in urn model. The first one is called a Splitting and Naming algorithm with high probability (S-N w.h.p.), i.e. the procedure is successful with probability $1 - \mathrm{o}\left(n^{-1}\right)$, as $n \to \infty$. The second one is referred as a Splitting and Naming algorithm with very high probability (S-N w.v.h.p.), i.e. for any $c \geqslant 1$, the procedure is successful with probability $1 - \mathrm{o}\left(n^{-c}\right)$, as $n \to \infty$. Names of those two algorithms suggest that they are reliable. Nevertheless, the exact rates of convergence were not provided in [79]. Hence we take this topic under investigation.

Both procedures consist of two phases of random draws, which will be presented in next sections.

General Splitting and Naming procedure

Let us explain an execution of the basic algorithm. At first, every device in the network independently draws bits uniformly at random ('0's and '1's) until the first occurrence of a bit '1'. Such the number of bits is called *a lifetime* of the node and is indicated by t_v. Naturally $t_v \sim \mathrm{Geo}\left(\frac{1}{2}\right) + 1$. After this phase vertices are splitted by theirs lifetimes into disjoint groups. Further, each vertex v chooses its *identity* id_v (a name) uniformly at random from some set of the form $[0 : f(t_v) - 1]$, where f is some given function, defined on $\mathbb{N}$. Finally, every agent saves its *label* as a couple (t_v, id_v). Realize that a set of all possible couples (labels) is naturally ordered by lexicographic order (see Section 1.2.3), so an every finite subset has a maximum (i.e. the reversed order is well on this set). Algorithm 3 presents a general base of Splitting and Naming election procedure. Note that it cannot be directly treated as a tossing framework of Algorithm 2. In order to provide a restricted version of Splitting and Naming

algorithm, firstly Algorithm 2 should be used with $D(\Theta) = \text{Geo}\left(\frac{1}{2}\right)$ and the provided L, and the second phase should run a version of Algorithm 2, which omits the lines 1 and 3 and further utilize $D(\Theta) = \text{Uni}\left([0 : f(t_v) - 1]\right)$ (here, there is no truncation needed, so in the second case the support of distribution is bounded per se. Therefore, the whole algorithm in beeping model is as presented

ALGORITHM 3: General base of Splitting and Naming draw

```
   procedure SelectLabel(f)                                  // For each node
1  |  generate G ~ Geo(1/2)
2  |  t_v = G + 1
3  |  generate a name id_v ~ Uni([0 : f(t_v) - 1])
4  |  label_v = (t_v, id_v)
```

in Algorithm 4. Note that one may rewrite Algorithm 4 in such the way that the bits of B_1 and B_2 are attained only when needed in "for" loops (i.e. utilized draws are performed lazily). Let us indicate, that standard versions of Splitting and Naming algorithms assume $L = \infty$. Nevertheless, for any $L \in \mathbb{N} \cup \{\infty\}$, this procedure is fully-anonymous.

Two types of Splitting and Naming procedures

Consider Algorithm 4. In this section, we assume that $p = \frac{1}{2}$. Moreover the first Splitting and Naming algorithm (S-N w.h.p.) provided in [79] utilizes the function $f_1(x) = x^3 2^x$ and a definition of the function of the second procedure (S-N w.v.h.p) involves $\lg^*$ function (see Section 1.2.6 for definition), namely it uses $f_2(x) = 2^{x \lg^*(x)}$.

The authors of [79] took notice, that $\lg^*$ can be exchanged with any other slowly growing function, however we are going to analyze only the original case.

Splitting and Naming frameworks naturally follow a lexicographic orders (see Section 1.2.3 for definition): firstly, fissured by lifetimes of nodes of the network, and secondly, by names of devices in each separated group (which consist of the nodes with the same lifetimes).

Realize that in both algorithms, the draw of lifetime t_v determines the number of bits that are necessary in order to save the name id_v in the Naming phase, hence according to approach via two runs of Algorithm 2, some of the devices (which are not the winner of Splitting phase) may resign to execute Naming phase, hence this algorithm may be interpreted as a quitting one.

Probability of failure for Splitting and Naming "with high probability"

As mentioned before, t_v follows $\text{Geo}\left(\frac{1}{2}\right) + 1$ distribution. Thence we are going to analyze random variables with the geometric distribution instead of the stopping time of bits-drawing process described at the start of Section 2.7.3. Without a loss of generality we attribute each vertex with different number from the set $[n]$.

ALGORITHM 4: General Splitting and Naming algorithm in Beeping Model of communication in Urn Model

```
   procedure Send(L, p, f)                                   // For each node
1  |  Leader = true
2  |  generate X ~ Geo(p)
3  |  msg_1 = min(X, L)
4  |  K_1 = ⌈lg(L + 1)⌉
5  |  B_1 = BIN_{K_1}(msg_1)
6  |  for i = K_1 − 1 down to 0 do
7  |  |  if [[B_1[i] = 1]] then
8  |  |  |  send BEEP
9  |  |  else
10 |  |  |  listen
11 |  |  |  if hear BEEP or collision then          // other node transmits
12 |  |  |  |  Leader = false
13 |  |  |  |  break
14 |  X ~ Uni([0 : f(msg_1 + 1) − 1])
15 |  msg_2 = X
16 |  K_2 = ⌈lg(f(msg_1 + 1))⌉
17 |  B_2 = BIN_{K_2}(msg_2)
18 |  for i = K_2 − 1 down to 0 do
19 |  |  if [[B_2[i] = 1]] then
20 |  |  |  send BEEP
21 |  |  else
22 |  |  |  listen
23 |  |  |  if hear BEEP or collision then          // other node transmits
24 |  |  |  |  Leader = false
25 |  |  |  |  break
```

Let $n \geqslant 2$ and assume that $G_1, G_2, \ldots G_n$ are i.i.d. random variables with distribution $\mathrm{Geo}\left(\frac{1}{2}\right)$. Moreover, assume that for every vertex $i \in [n]$, $U_i \sim \mathrm{Uni}\left([0 : 2^{g_i+1}(g_i+1)^3 - 1]\right)$, where $(g_1, g_2, \ldots, g_n)$ is a realization of the random vector $(G_1, G_2, \ldots, G_n)$. All variables G_i and $U_j|G_j$, for $i, j \in [n]$, are independent.

Let us find some constraints on a failure of the S-N w.h.p. procedure. Let $F_{n,S-Nw.h.p.}$ denotes the event when at least 2 of n agents draw the same, maximal lifetime (amongst all the competitors) and have the identical, maximal names amongst all vertices with the same lifetimes, when using $f(x) = x^3 2^x$. $F_{n,S-Nw.h.p.}$ is then an event which represents failure during the execution of Splitting and Naming w.h.p. procedure. For simplification let us denote a set of all nodes, which choose the lifetime m by V_m.[40] A particular instance of the failure take place when all devices belong to V_1 and there are at least 2 vertices, which choose 1 in the second phase (then $(U_i|G_i = 1) \sim \mathrm{Uni}\left(\{0, 1\}\right)$ for each $i \in [n]$). Let us investigate an another case — when 2 realize the maximum amongst lifetimes of the processors in the Splitting phase. Then contestants from V_2 use $\mathrm{Uni}\left([0 : 2^5 - 1]\right)$ distribution in the second part of the algorithm to resolve the problem. Intuitively, this option, compared to the previous one, creates more opportunity to choose different identities, especially it is more probable to reach the unique maximum. Naturally this trend is satisfied in general — as the maximal lifetime gets bigger, then it is easier to choose a unique maximum in the second draw. We would like to find a lower bound on $\Pr[F_{n,S-Nw.h.p.}]$, so we can, for instance, consider only two first aforementioned situations. However, let us commence with the probability bounded only by the first described case (with $n > 1$):

$$\Pr[F_{n,S-Nw.h.p.}] \geqslant \Pr\left[\bigwedge_{i=1}^{n} G_i = 1, \neg(\exists!\ k)\ U_k = 1\right] = 2^{-n}(1 - n2^{-n}) \geqslant 2^{-(n+1)}.$$

Let us define $I_1(n) := 2^{-n}(1 - n2^{-n})$.

It is obvious that $B_1(n)$ is quite essential for small n, so this procedure works inaccurate, until n is big. Sometimes, in Ad Hoc networks, small sub-networks are created in order to struggle with some assignments,

Now, let us consider an exact probability of the collision for $n = 2$ for S-N w.h.p. via polylogarithm[41]:

$$\begin{aligned}\Pr[F_{2,S-Nw.h.p}] &= \Pr[G_1 = G_2, U_1 = U_2] \\ &= \sum_{t=1}^{\infty} \left(\frac{1}{2}\right)^{2t} \frac{1}{2^t t^3} = \mathrm{Li}_3\left(\frac{1}{8}\right) = 0.12703\ldots\ .\end{aligned}$$

As we can clearly see, Splitting and Naming algorithm w.h.p. fails once in 8 tries when there are only two competitors, what is absolutely unacceptable from a practical point of view.

[40] It is a random set.

[41] For a definition of polylogarithm see Section 1.2.11.

An analogous, slightly complicated calculation can be done for $n = 3$. Then the adequate probability of the failure can be divided as follows:

$$\begin{aligned}\Pr[F_{3,S-Nw.h.p}] =& \Pr[G_1 = G_2 = G_3, U_1 = U_2 = U_3] \\ &+ 3\Pr[G_1 = G_2 = G_3, U_1 = U_2 > U_3] \\ &+ 3\Pr[G_1 = G_2 > G_3, U_1 = U_2]\end{aligned}$$

We are going to calculate each part separately. First of all

$$\begin{aligned}\Pr[G_1 = G_2 = G_3, U_1 = U_2 = U_3] &= \sum_{t=1}^{\infty} \left(\frac{1}{2}\right)^{3t} \left(\frac{1}{2^t t^3}\right)^2 \\ &= \sum_{t=1}^{\infty} \left(\frac{1}{2}\right)^{5t} \frac{1}{t^6} = \mathrm{Li}_6\left(\frac{1}{32}\right) .\end{aligned}$$

Moreover

$$\begin{aligned}\Pr[G_1 = G_2 = G_3, U_1 = U_2 > U_3] &= \sum_{t=1}^{\infty} \left(\frac{1}{2}\right)^{3t} \sum_{i=2}^{2^t t^3} \left(\frac{1}{2^t t^3}\right)^2 \frac{i-1}{2^t t^3} \\ &= \sum_{t=1}^{\infty} \left(\frac{1}{2}\right)^{6t+1} \frac{1}{t^9} \left(2^t t^3 - 1\right)\left(2^t t^3\right) \\ &= \sum_{t=1}^{\infty} \left(\frac{1}{2}\right)^{4t+1} \frac{1}{t^3} - \sum_{t=1}^{\infty} \left(\frac{1}{2}\right)^{5t+1} \frac{1}{t^6} \\ &= \frac{1}{2}\mathrm{Li}_3\left(\frac{1}{16}\right) - \frac{1}{2}\mathrm{Li}_6\left(\frac{1}{32}\right) .\end{aligned}$$

Similarly, we can calculate the latter term:

$$\begin{aligned}\Pr[G_1 = G_2 > G_3, U_1 = U_2] &= \sum_{t=2}^{\infty} \left(\frac{1}{2}\right)^{2t} \left(1 - \left(\frac{1}{2}\right)^{t-1}\right) \frac{1}{2^t t^3} \\ &= \sum_{t=2}^{\infty} \left(\frac{1}{2}\right)^{3t} \frac{1}{t^3} - \sum_{t=2}^{\infty} \left(\frac{1}{2}\right)^{4t-1} \frac{1}{t^3} \\ &= \mathrm{Li}_3\left(\frac{1}{8}\right) - \frac{1}{8} - 2\mathrm{Li}_3\left(\frac{1}{16}\right) + \frac{1}{8} \\ &= \mathrm{Li}_3\left(\frac{1}{8}\right) - 2\mathrm{Li}_3\left(\frac{1}{16}\right) .\end{aligned}$$

Therefore

$$\Pr[F_{3,S-Nw.h.p}] = 3\mathrm{Li}_3\left(\frac{1}{8}\right) - 4.5\mathrm{Li}_3\left(\frac{1}{16}\right) - 0.5\mathrm{Li}_6\left(\frac{1}{32}\right) = 0.0819669\ldots ,$$

so when $n = 3$, the algorithm falls short approximately once every 12 runs, which is still unacceptable.

Probability of failure for Splitting and Naming "with very high probability"

It can be shown that the second algorithm from [79] (S-N w.v.h.p.) behaves even worse than the first one, when there are only two nodes. Let $F_{n,S-Nw.v.h.p}$ denotes the event when at least 2 of n devices draw the same, maximal lifetime (amongst all the competitors) and have the identical, maximal names amongst all vertices with the same lifetimes, when $f(x) = 2^{x \lg^*(x)}$, what can be identified as the failure of Splitting and Naming algorithm w.v.h.p.

Then

$$\Pr[F_{2,S-Nw.v.h.p.}] = \sum_{t=1}^{\infty} \frac{1}{2^{t \lg^*(t)}} \left(\frac{1}{2}\right)^{2t} > \sum_{t=1}^{4} \frac{1}{2^{3t}} = 0.14284\ldots\ ,$$

what is indeed bigger than $\Pr[F_{2,S-Nw.h.p.}] = 0.12703\ldots$ (and we only included first four expressions of the series).[42] On the other hand

$$\Pr[F_{2,S-Nw.v.h.p.}] = \Pr[G_1 = G_2, Y_1 = Y_2] < \sum_{n=1}^{\infty} \left(\frac{1}{2}\right)^{3n} = \frac{1}{7} = 0.14286\ldots\ ,$$

so $0.14284 < \Pr[F_{2,S-Nw.v.h.p.}] < 0.14287$.

Let us use the same assumptions and notations as previously (for S-N w.h.p). Additionally, let now $\Upsilon_i \sim \mathrm{Uni}\left([0 : 2^{g_i \lg^*(g_i)} - 1]\right)$, where for $i \in [n]$, g_i are realizations of G_i. Naturally we assume that all G_i and $\Upsilon_i | G_i$ are independent.

We are going to find a precise lower bound of $\Pr[F_{n,S-Nw.v.h.p.}]$. When a node draws 2 in the first phase, then in the next one it draws according to $\mathrm{Uni}\left([0 : 2^2 - 1]\right)$ distribution (instead of $\mathrm{Uni}\left([0 : 2^5 - 1]\right)$, as before). This fact provoke additional collisions with substantial probability, so we decided to investigate this particular case as well, when looking for the tolerable lower boundary. Namely, we add the case, when some of devices pick 2 at first and the maximum is not unique in the latter draw. The above argument convinces, that, for small n, an accuracy of S-N w.v.h.p. is worse than for S-N w.h.p., conversely to theirs names. Beneath we calculate the appropriate limitation for the failure's probability:

$$\Pr[F_{n,S-Nw.v.h.p.}] \geqslant \Pr\left[\bigwedge_{i=1}^{n} G_i = 1, \neg(\exists!\ s)\Upsilon_s = 1\right] +$$

$$+ \sum_{k=2}^{n} \Pr\left[|V_2| = k, |V_1| = n - k, \neg(\exists!\ s \in V_2) \max_{i \in V_2}\{\Upsilon_i\} = \Upsilon_s\right]$$

$$= I_1(n) + \sum_{k=2}^{n} \binom{n}{k} 2^{-(n-k)} 2^{-2k} \left(1 - \frac{k}{4} 4^{-k+1} - \frac{k}{4} 2^{-k+1} - \frac{k}{4}\left(\frac{3}{4}\right)^{k-1}\right) = \ldots$$

[42] One can change slightly the definition of $\lg^*$ to increase the reliability in exchange for a one bit of memory. It is also possible to utilize other slowly growing function as was suggested in [79].

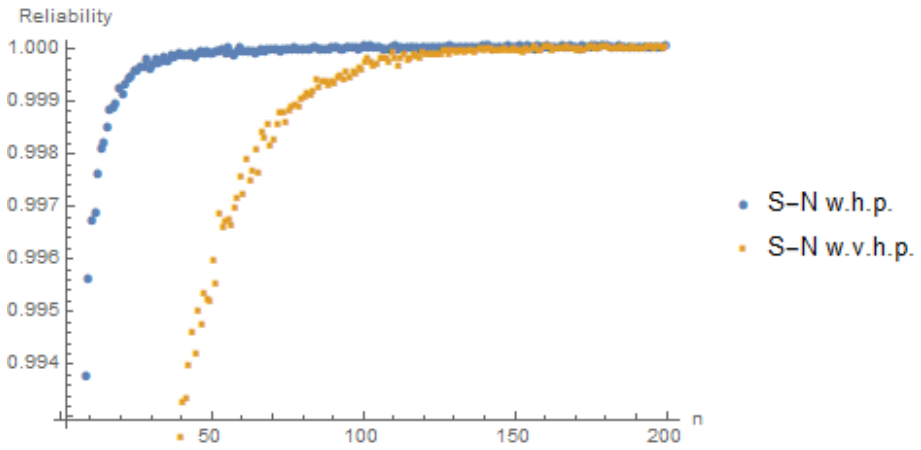

Figure 2.4: Comparison of simulated probabilities of success of the Splitting–Naming algorithms with high probability (blue dots) and with very high probability (orange squares), depending on the number of devices $n \in [200]$. For each n we executed Monte Carlo algorithms with 10^5 experiments and obtained the average probabilities of success.

$$\begin{aligned}
\ldots &= 2^{-n}\left[1 - n2^{-n} + \sum_{k=2}^{n}\binom{n}{k}2^{-k}\left(1 - k4^{-k} - \frac{k}{2}2^{-k} - \frac{k}{3}\left(\frac{4}{3}\right)^{-k}\right)\right] \\
&\overset{\text{Fact } 1.2.6}{=} 2^{-n}\left\{1 - n2^{-n} + \left[\left(\frac{3}{2}\right)^{n} - \frac{n}{2} - 1\right] - \frac{n}{8}\left[\left(\frac{9}{8}\right)^{n-1} - 1\right]\right. \\
&\qquad \left. - \frac{n}{8}\left[\left(\frac{5}{4}\right)^{n-1} - 1\right] - \frac{n}{8}\left[\left(\frac{11}{8}\right)^{n-1} - 1\right]\right\} \\
&= 2^{-n}\left[-n2^{-n} + \left(\frac{3}{2}\right)^{n} - \frac{n}{8}\left(1 + \left(\frac{9}{8}\right)^{n-1} + \left(\frac{5}{4}\right)^{n-1} + \left(\frac{11}{8}\right)^{n-1}\right)\right] .
\end{aligned}$$

Let $S_{n,\text{alg}}$ denotes the event opposite to $F_{n,\text{alg}}$, where $\text{alg} \in \{S{-}Nw.h.p., S{-}Nw.v.h.p.\}$. In Figure 2.4 we present a comparison of $\Pr[S_{n,S-Nw.h.p}]$ and $\Pr[S_{n,S-Nw.v.h.p.}]$ with respect to n ranged in $[200]$, where each of the probabilities is simulated as Monte Carlo algorithm with 10^5 repetitions of Leader Election experiments. A slight surprise emerges for $n < 200$, when the theoretically more reliable algorithm performs even worse. Figure 2.4 shows that Splitting and Naming algorithms behave properly for a large number of nodes (authors studied theirs asymptotic maintenance), but they behave unsatisfactorily for a small number of nodes.

From Figure 2.4 it is difficult to judge, when the hypothetically better algorithm begins (with respect to the number of contestants) to be more reliable.

However, note that $2^{x \lg^*(x)} > x^3 2^x$ whenever $x > 10$ (we only consider $x \in \mathbb{N}$). However, from Fact 1.3.5, we get that $\Pr[M > 10] = 1 - \left(1 - \frac{1}{2^{10}}\right)^n$, where $M \sim \mathrm{MGeo}\left(n, \frac{1}{2}\right)$. Especially, when $n = 2^9 = 512$, $\Pr[M > 10] \approx 1 - e^{-\frac{1}{2}} = 0.393469\ldots$, so for $n \leqslant 512$ devices, S-N w.v.h.p. performs worse than S-N w.h.p.

Number of rounds

Consider a sequence of i.i.d. random variables $(G_i)_{i=1}^n$ with $\mathrm{Geo}\left(\frac{1}{2}\right)$ distribution and let $M_n = \max(G_1, G_2, \ldots, G_n)$.

We would like to find some restriction on a number of bits K to save a maximum identifier drawn during the S-N w.h.p. Let m_n be realization of M_n variable (the maximum of all lifetimes) during the first phase of the algorithm. Then the second phase will be conducted according to $\mathrm{Uni}\left([0 : 2^{m_n+1}(m_n+1)^3 - 1]\right)$ distribution. To save such the identifier, we need $\lceil \lg m_n + 1 \rceil$ bits for the first one and $m_n + 1 + \lceil 3 \lg m_n + 1 \rceil$ bits for the latter one.

By Fact 1.3.5, $\Pr[M_n > k] = 1 - \left(1 - \left(\frac{1}{2}\right)^k\right)^n$. In a case, when $n \ll 2^k$, we may use inequality $\Pr[M_n > k] > n2^{-k} - \binom{n}{2}2^{-2k}$ (see Fact 1.2.6). Let $0 < \varepsilon < 1$. If we assume $n2^{-k} = \varepsilon$, then $\Pr[M_n > \lg(n) - \lg(\varepsilon)] > \varepsilon(1 - \frac{\varepsilon}{2})$. Therefore at least approximately $\lg(n) - \lg(\varepsilon) + 4\lg(\lg(n) - \lg(\varepsilon))$ bits are needed with probability close to ε to save the biggest lifetime. If we would like to provide restricted, Atlantic City version of S-N w.h.p. with $(1-\varepsilon)$-reliability, then we need at least the aforementioned number of bits, which is strictly greater than the respective parameter of ULE algorithm (see Equation 2.19). It worth to realize that Splitting and Naming procedure was designed as fully-anonymous algorithm, however, the above argument shows that the sizes of labels of some of the devices grows approximately logarithmic with respect to the number of all devices. Moreover, note that in the S-N w.v.h.p., with the similar assumptions we obtain, that approximately $\lg(n) - \lg(\varepsilon) + \lg(\lg(n) - \lg(\varepsilon)) + \lg^*(\lg(n) - \lg(\varepsilon))$ bits are needed, which is still greater than for ULE algorithm. One can compare that when $\varepsilon < 2^{-5}$, then the restricted, Atlantic City version of S-N w.v.h.p. needs more rounds than the restricted Atlantic City two-phases version of LGE, provided before.[43] This section makes one aware that Splitting and Naming algorithm provided in [79] **should not** be used in other arrangements than Las Vegas and fully-anonymous. It can be utilized when one do not care about the memory of devices and whenever it is used, then the number of devices should be enormous with high probability.

It worth to mention that Splitting and Naming with very high probability is sometimes treated as the state-of-the-art of Leader Election algorithms. For instance, a well known paper about Leader Election in radio networks with Signal-to-Interference-plus-Noise-Ratio (SINR) model of communication was investigated by M.M. Halldórsson et al. in [57] in 2019. Theirs solution utilized Splitting and Naming algorithm with $f(x) = x^4 2^x$, which is slightly more reliable than S-N w.h.p., but more energy is needed to send the information about

[43] This fact is irrespective of the number of devices.

the identifiers. Surprisingly, the solution of M.M. Halldórsson et al. was created independently of the one from [79].

2.8 Geometric Green Leader Election

2.8.1 Introduction

The name of this section refers to already mentioned (in Section 2.7.2) Leader Green Election algorithm proposed by P.Jacquet et al. in [64]. In Section 2.7.2 we have seen, that some of the parameters of LGE algorithm can be changed in order to provide more efficient solution and also that restricted version of LGE can be utilized as well. We would like to delve into the idea of Leader Elections in urn model according to restricted geometric distribution $\mathrm{Geo}\,(p, L)$ in order to improve the LGE procedure to Geometric Green Leader Election algorithm (further abbreviated to GeoGLE algorithm) with judicious bounds on the failure's probability and the time complexity.

Namely, we are going to discuss a universal, fair, quasi-anonymous, oblivious-quitting, $(1-\varepsilon)$-reliable Atlantic City LEA in single-hop beeping model without collision detection, in urn model according to distribution $\mathrm{Geo}\,(p, L)$. For a given capacity of the network $N \in \mathbb{N}$ and mistake frequency $\varepsilon > 0$, we would like to determine such parameters p and L which guarantee that the considered algorithm will be $(1-\varepsilon)$-reliable for an arbitrary number of nodes $n \in [N]$.

Our main goal is to find such the algorithm that optimize the number of necessary rounds K to achieve the required reliability. Specifically we limit ourselves to a class of GeoGLE algorithms described before.

2.8.2 GeoGLE Algorithm

In GeoGLE algorithm, each node generates independently its identity according to $\mathrm{Geo}\,(p, L)$ distribution. In the terms of Algorithm 2 it can be simply done by putting $D(\Theta) = \mathrm{Geo}\,(p)$. Since the support of geometric distribution is defined to be $\mathbb{N}_0$, there may some consternation about L parameter in restriction operation, so Algorithm 5 presents a pseudo-code of the aforementioned GeoGLE algorithm in the arrangement of Algorithm 2.

Properties

Now we discuss basic properties of GeoGLE algorithm. Especially, we will show how to choose such parameters p and L that GeoGLE algorithm ends successfully with controlled probability.

Similarly to the notation from Section 2.6, let $S_{n,\mathrm{Geo}(p,L)}$ denotes the following event: "if random variables $X_1, \ldots, X_n$ are i.i.d. with $\mathrm{Geo}\,(p, L)$ distribution, then $\mathrm{card}\,(\{i : X_i = \max\{X_j : j \leqslant n\}\}) = 1$". In other words, $S_{n,\mathrm{Geo}(p,L)}$ holds if our GeoGLE algorithm successfully elects a leader when applied to a network where n nodes compete.

ALGORITHM 5: Beeping Leader Election Algorithm

```
   procedure Select(p, L)                                  // For each node
1  |  Leader = true
2  |  generate G ~ Geo(p)
3  |  X = min(G, L)
4  |  K = ⌈lg(L + 1)⌉
5  |  B = BIN_K(X)
6  |  for i = K − 1 down to 0 do
7  |  |  if [[B[i] = 1]] then
8  |  |  |  Send BEEP
9  |  |  else
10 |  |  |  listen
11 |  |  |  if hear BEEP or collision then        // other node transmits
12 |  |  |  |  Leader = false
13 |  |  |  |  break
```

Let us signal that the beneath fact is a result of application of classical formulas for cubic equations:

Fact 2.8.1. *If* $c \in (0,4)$ *and* $\varepsilon = \frac{c^2}{(c+2)^3}$, *then*

$$\begin{aligned} c = &-\frac{\left(1-i\sqrt{3}\right)\sqrt[3]{54\varepsilon^2+6\sqrt{3}\sqrt{27\varepsilon^4-2\varepsilon^3}-18\varepsilon+1}}{6\varepsilon} \\ &+\frac{\left(1+i\sqrt{3}\right)(12\varepsilon-1)}{6\varepsilon\sqrt[3]{54\varepsilon^2+6\sqrt{3}\sqrt{27\varepsilon^4-2\varepsilon^3}-18\varepsilon+1}} \\ &-\frac{6\varepsilon-1}{3\varepsilon}\,, \end{aligned}$$

where i *is an imaginary unit.*

A main contribution of this part is Theorem 2.8.1. Its formulation includes two branches of W-Lambert function (see Section 1.2.12).

Theorem 2.8.1. *Suppose that* $0 < \varepsilon < \frac{2}{27}$ *and* $N \in \mathbb{N}$.[44] *Moreover, let*

1. $c \in (0,4)$ *be such that* $\varepsilon = \frac{c^2}{(c+2)^3}$,
2. $C(N,\varepsilon) = (1+\sqrt{(c+2)\varepsilon})\frac{(c+2)\varepsilon}{2(N-1)}$
3. $\tilde{C}(N,\varepsilon) = \frac{1+\sqrt{\frac{6N-6}{N+4}\frac{2}{27}}}{2}\varepsilon$,
4. $\tilde{\varepsilon} = \varepsilon - \min(C(N,\varepsilon), \tilde{C}(N,\varepsilon))$,

[44] Let us mark, that the case of small N is slightly problematic, especially when ε is close to $\frac{2}{27}$. This is the usual case when $\tilde{C}(N,\varepsilon)$ is necessary. Nevertheless, when one does not want to use this formula, it is sufficient to choose slightly bigger N.

5. $\tau = 1 - \frac{\ln(1-3\varepsilon)}{3}$,

6. $K = \left\lceil \lg \left(1 - \frac{W_{-1}\left(-\frac{2\tilde{\varepsilon}}{e^2(N-1)^2} \right)}{4\tilde{\varepsilon}} \right) \right\rceil$,

7. $L = 2^K - 1$,

8. $p = 1 - \left(\frac{-2W_0\left(-\sqrt{\frac{\tau}{8L}}\right)}{N-1} \right)^{\frac{1}{L}}$.

Assume that moreover $\lambda_N := (N-1)(1-p)^L < \sqrt{(c+2)\varepsilon}$. *Then for every* $n \in [N]$ *we have* $\Pr\left[S_{n,\mathrm{Geo}(p,L)}\right] \geqslant 1 - \varepsilon$.

Note that c (provided explicitly in Fact 2.8.1) and τ depend only on parameter ε. Moreover $C(N,\varepsilon)$, $\tilde{C}(N,\varepsilon)$ and $\tilde{\varepsilon}$ depend only on the initial parameters N and ε. These five auxiliary notations are designated to present the thesis of main part of the Theorem 2.8.1 and its proof in much simpler form. One can wonder if $\tilde{\varepsilon} > 0$. However, in Appendix B, we will show Corollary 10, which states that $\tilde{\varepsilon} > \frac{\varepsilon}{6}$.[45] The sixth emphasized formula in Theorem 2.8.1 let us attain the number of the rounds K for Algorithm 5. Let us remark, that we will show that one can provide $(1-\varepsilon)$-reliable GeoGLE algorithm, if the number of atoms of distribution L used in GeoGLE algorithm is at least $\left\lceil -\frac{W_{-1}\left(-\frac{2\tilde{\varepsilon}}{e^2(N-1)^2} \right)}{4\tilde{\varepsilon}} \right\rceil$. This let us choose different L, however we should bear in mind that an increase of L gives an additional opportunity to avoid collisions (similar argument was provided for non-anonymous model in the proof of Lemma 3), but on the other side, it can eventually raise the number of needed bits K. As long as we do not want to exceed provided minimal number of rounds K, we should choose L of the form $2^K - 1$ (see line 4 of Algorithm 5). This is why Theorem 2.8.1 firstly provide a formula for K with respect to parameters N and ε and then assume $L = 2^K - 1$. We also should remember that the value of K provided in Theorem 2.8.1 may not be optimal for a given N and ε parameters, so in some cases, there may be possible to provide $(1-\varepsilon)$-reliable LEA designated for the network with capacity N, which uses slightly less bits. The last itemized point of the Theorem 2.8.1 is to assume such parameter p that the distribution $\mathrm{Geo}(p,L)$ will guarantee $(1-\varepsilon)$-reliability of GeoGLE algorithm. For the sake of clarity, a lengthy, technical proof of Theorem 2.8.1, together with miscellaneous remarks which emerge from the proof, are postponed to Appendix B. Let us only mention that the assumption about λ_N in the formulation of Theorem 2.8.1 is a technical condition, which is unavoidable, when one wants to provide any $(1-\varepsilon)$-reliable GeoGLE algorithm, when $\varepsilon < \frac{2}{27}$. This notation

[45] This inequality is inefficient. Usually $\tilde{\varepsilon} \approx \varepsilon$.

will be highly exploited in Appendix B. As we have highlighted, the formulation of Theorem 2.8.1 utilize two different real branches of W-Lambert function. However, let us recall that in majority of program languages and mathematical packages those functions are implemented and also can be easily approximated (see Section 1.2.12).

2.8.3 Discussion

Let us consider the arrangement of parameters of GeoGLE algorithm provided in Theorem 2.8.1. Remark that formula (1.10) provides

$$-W_{-1}\left(-\frac{2\tilde{\varepsilon}}{e^2(N-1)^2}\right) \approx 2 + 2\ln(N-1) - \ln(2\tilde{\varepsilon}) \ .$$

Therefore our algorithm runs in constant number[46] of approximately

$$K \approx \lg\left(1 + \ln\frac{N}{\sqrt{2\tilde{\varepsilon}}}\right) + \lg\left(\tilde{\varepsilon}^{-1}\right) - 1 = \lg\left(\ln\frac{eN}{\sqrt{2\tilde{\varepsilon}}}\right) + \lg\left(\tilde{\varepsilon}^{-1}\right) - 1 \qquad (2.20)$$

rounds in single-hop arrangement of beeping model. Remark that the term $\lg\left(\varepsilon^{-1}\right)$ is unavoidable (see the discussion in Section 2.6.4 and the result for the simpler model in Section 2.5.6) in the general model of oblivious LEAs. Note also that $\lg(\ln(N))$ term grows very slow as N become very large. Therefore Geometric Green Leader Election Algorithm fits in the reasoning of P. Jacquet et al. [64], who have described theirs algorithm with an adjective "Green", connoted with low energy solutions. Nevertheless, let us highlight that usually in practise we are not interested in the asymptotic properties of algorithm.[47]

In Table 2.1 we present examples of parameters p for GeoGLE algorithm obtained from Theorem 2.8.1 for $\varepsilon \in \{10^{-6}, 10^{-9}\}$ and some arbitrary network capacities N. Realize that in all cases in Table 2.1, p is relatively close to ε. Moreover, in Appendix B we show that the parameter p from Theorem 2.8.1 satisfies $p < 3\varepsilon$.

Basing on these remarks, one may be likely to assume that $p = \varepsilon$ naively. On the other hand Figure 2.5 shows that this hypothesis may have relatively big impact on the probability of failure and should be rejected. Basing on Figure 2.5, we can also guess, that it is relatively careful to slightly overestimate the parameter p.

Example 9. *Indeed, assume that $n = 200$ devices in the network compete in the election using GeoGLE algorithm. Let us consider the case when $N = 200$ and $\varepsilon = 0.001$. Then, from Theorem 2.8.1 we obtain $K = 13$ and $p = 0.0012372817\ldots$. In this case one can we obtain that $\Pr\left[S_{200,\mathrm{Geo}(\varepsilon,2^{13}-1)}\right] = 0.9980388299\ldots \approx 1 - 2\varepsilon$, but $\Pr\left[S_{200,\mathrm{Geo}(p,2^{13}-1)}\right] = 0.9993504638\ldots > 1 - \varepsilon$. This show that solution proposed in Theorem 2.8.1 meets the assumptions of*

[46] Since N and ε are given a priori

[47] Section 2.7.3 emphasized dangers, which may arise when one focus only on such the properties.

	$\varepsilon = 10^{-6}$			$\varepsilon = 10^{-9}$		
N	$\tilde{\varepsilon}$	K	p	$\tilde{\varepsilon}$	K	p
10	$8.88574...\cdot 10^{-7}$	23	$1.25347...\cdot 10^{-6}$	$8.88879...\cdot 10^{-10}$	33	$1.62757...\cdot 10^{-9}$
20	$9.47219...\cdot 10^{-7}$	23	$1.34254...\cdot 10^{-6}$	$9.47364...\cdot 10^{-10}$	33	$1.71456...\cdot 10^{-9}$
50	$9.79534...\cdot 10^{-7}$	23	$1.45548...\cdot 10^{-6}$	$9.79590...\cdot 10^{-10}$	33	$1.82485...\cdot 10^{-9}$
100	$9.89870...\cdot 10^{-7}$	23	$1.53932...\cdot 10^{-6}$	$9.89898...\cdot 10^{-10}$	34	$9.73534...\cdot 10^{-10}$
200	$9.94961...\cdot 10^{-7}$	23	$1.62255...\cdot 10^{-6}$	$9.94974...\cdot 10^{-10}$	34	$1.01417...\cdot 10^{-9}$
500	$9.97990...\cdot 10^{-7}$	23	$1.73214...\cdot 10^{-6}$	$9.97996...\cdot 10^{-10}$	34	$1.06768...\cdot 10^{-9}$
1000	$9.98996...\cdot 10^{-7}$	23	$1.81489...\cdot 10^{-6}$	$9.98999...\cdot 10^{-10}$	34	$1.10809...\cdot 10^{-9}$
2000	$9.99498...\cdot 10^{-7}$	23	$9.69449...\cdot 10^{-7}$	$9.99500...\cdot 10^{-10}$	34	$1.14846...\cdot 10^{-9}$
5000	$9.99799...\cdot 10^{-7}$	24	$1.02408...\cdot 10^{-6}$	$9.99800...\cdot 10^{-10}$	34	$1.20182...\cdot 10^{-9}$
10000	$9.99900...\cdot 10^{-7}$	24	$1.06540...\cdot 10^{-6}$	$9.99900...\cdot 10^{-10}$	34	$1.24217...\cdot 10^{-9}$
20000	$9.99950...\cdot 10^{-7}$	24	$1.10672...\cdot 10^{-6}$	$9.99950...\cdot 10^{-10}$	34	$1.28252...\cdot 10^{-9}$
50000	$9.99980...\cdot 10^{-7}$	24	$1.16134...\cdot 10^{-6}$	$9.99980...\cdot 10^{-10}$	34	$1.33586...\cdot 10^{-9}$
100000	$9.99990...\cdot 10^{-7}$	24	$1.20265...\cdot 10^{-6}$	$9.99990...\cdot 10^{-10}$	34	$1.37620...\cdot 10^{-9}$

Table 2.1: Parameters p nad K provided for $\varepsilon \in \{10^{-6}, 10^{-9}\}$ and several selected values of capacity N, established according to Theorem 2.8.1.

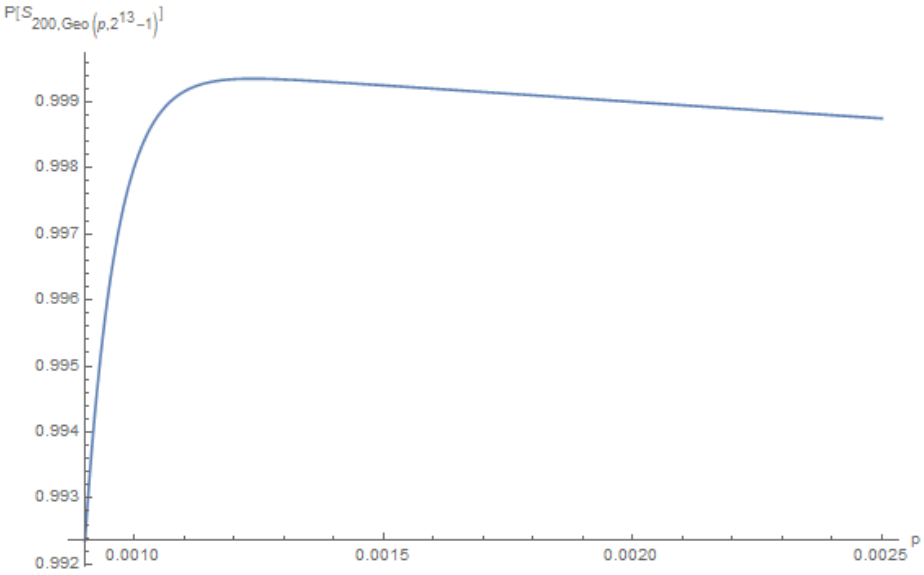

Figure 2.5: Plot of exact probabilities of success of Geometric Green Leader Election for $n = 200$, $\varepsilon = 0.001$, $K = 13$ for $p \in [0.0009, 0.0025]$.

LEA in contrast to the solution with $p = \varepsilon$. Moreover, a maximal probability of success using GeoGLE algorithm for $n = 200$ and $K = 13$ can be found numerically and it equals $\Pr\left[S_{200,\mathrm{Geo}(p_{max},2^{13}-1)}\right] = 0.99935046444\ldots$, where $p_{max} = 0.0012376712\ldots$.
One can also easily see that

$$\Pr\left[S_{200,\mathrm{Geo}(p_{max},2^{13}-1)}\right] - \Pr\left[S_{200,\mathrm{Geo}(p,2^{13}-1)}\right] = 6.067879\ldots\cdot 10^{-10} < \varepsilon^3 \ ,$$

what affirms the effectiveness of solution proposed in Theorem 2.8.1.

Now let us choose a bigger number of admissible devices in the rivalry e.g. $N = 1000$. Then with the same number of stations $n = 200$ and the same ε, we attain $p = 0.00143399705\ldots$ and $K = 13$. Again by Theorem 2.8.1, it occurs that $\Pr\left[S_{200,\mathrm{Geo}(p,2^{13}-1)}\right] = 0.99928160414\ldots > 1 - \varepsilon$. Naturally, the bigger value of N has negative influence on the choice of p parameter, because it should be adequate for a wider class of networks. Consequently and naturally it has negative influence on the probability of success as well.

The aforementioned examples get attention to a fact that the correct choice of the parameter p very subtly depends on the parameters N and ε and confirm the usability and quality of Theorem 2.8.1.

In Figure 2.6 we present a plot of failure's probability $\Pr\left[F_{100,\mathrm{Geo}(p,2^{23}-1)}\right]$ (q_{50}, depicted as an orange polyline) for the arrangement of Leader Election according to $\mathrm{Geo}\left(p, 2^{23}-1\right)$ distribution for $n = 100$ devices.[48] According

[48] See e.g. [86] for an introductory to interval estimation theory.

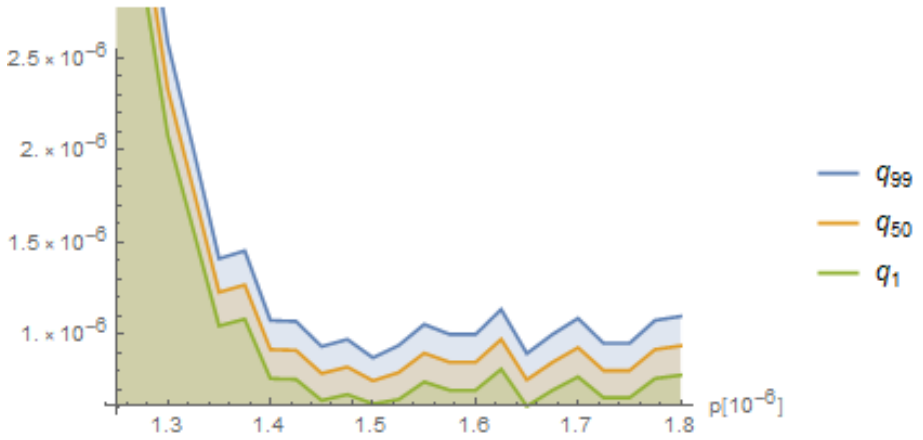

Figure 2.6: Plot of $1 - \Pr\left[S_{100,\mathrm{Geo}(p,2^{23}-1)}\right]$ (denoted by q_{50}) together with respective confidence intervals ($[q_1, q_{99}]$) at confidence level 0.98, simulated by Monte Carlo simulations with $2{\cdot}10^8$ repetitions for different p from $[1.3{\cdot}10^{-6}, 1.8{\cdot}10^{-6}]$ with stride $0.025 \cdot 10^{-6}$.

to Theorem 2.8.1, in GeoGLE with initial parameters $N = 100$ and $\varepsilon = 10^{-6}$, we should use Geo $\left(1.5393203\ldots \cdot 10^{-6}, 2^{23} - 1\right)$ distribution. The smallest estimated value of failure's probability (amongst presented in Figure 2.6) is obtained for $p = 1.5{\cdot}10^{-6}$ and equals $7.5{\cdot}10^{-7}$. For comparison, we have conducted similar Mote Carlo experiment for $p = 1.5393203\ldots{\cdot}10^{-6}$, what gave the estimation of $\Pr\left[F_{100,\mathrm{Geo}(1.5393203\ldots\cdot 10^{-6},2^{23}-1)}\right]$ as $6.8 \cdot 10^7$, which is smaller than the one for $p = 1.5 \cdot 10^{-6}$. This experiment affirms the quality of parameters provided in Theorem 2.8.1. One can easily see that, when $p < 1.4 \cdot 10^{-6}$, then probability of failure harshly grows as p parameters gets smaller. For instance, the smallest parameter p from the plot give an estimation of $\Pr\left[F_{100,\mathrm{Geo}(1.25\cdot 10^{-6},2^{23}-1)}\right]$ as $4.51 \cdot 10^{-6} \approx 4.5\varepsilon$. Moreover, similar experiment for $p = \varepsilon$ provided estimation of $\Pr\left[F_{100,\mathrm{Geo}(10^{-6},2^{23}-1)}\right]$ as $0.00025094 > 250\varepsilon$. This confirms our previous speculation from Example 9 that we should not use naively $p = \varepsilon$.

Apart from the failure's probabilities per se, in Figure 2.6 we have also presented confidence intervals of particular estimators, at confidence level 0.98. q_i denotes the approximation of the i-th quantile of the distribution of the estimator of the probability of failure. According to de Moivre—Laplace Theorem 1.3.2, q_i (for $i \in [100]$) can be approximated by $\mu + \Phi\left(\frac{i}{100}\right)\sqrt{\frac{\mu(1-\mu)}{\mathrm{reps}}}$, where μ is the average number of simulated failures, Φ is the cumulative distribution function of standard normal distribution $\mathcal{N}(0,1)$ and reps is the number of repetitions of Monte Carlo algorithm. Therefore, for each parameter p, $\Pr\left[F_{100,\mathrm{Geo}(p,2^{23}-1)}\right] \in [q_1(p), q_{99}(p)]$ (between green and blue polylines) with probability ~ 0.98. Basing on Figure 2.6, it is difficult to judge, which parameter p is the best in this case. For instance $p = 1.5{\cdot}10^{-6}$ or $p = 1.65{\cdot}10^{-6}$ seem all

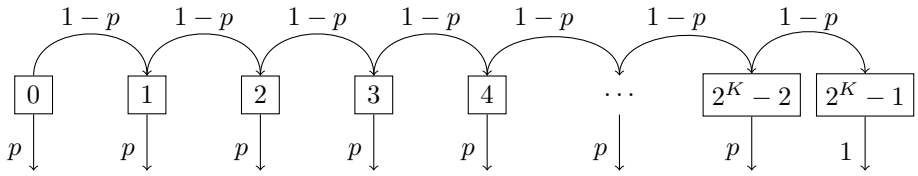

Figure 2.7: Graphical representation of standard method of drawing a realization of a random variable with Geo $\left(p, 2^K - 1\right)$ distribution. Arrows from the urns (depicted as squares) directed downwards denote, that the appropriate value is chosen.

right. However, Figure 2.6 shows that it is quite probable, that $p = 1.625 \cdot 10^{-6}$ is insufficient. Therefore, convinced by the whole experiment, we highly recommend, to use p provided in Theorem 2.8.1.

Let us mention that in Section B.5 we will try to answer the question — how much should we increase the capacity of the network N in order to increase K by 1? It shows that expanding the memory a little bit, usually allows to exceed significantly the capacity of the network provided a priori with a preservation of $(1 - \varepsilon)$-reliability regime.

2.8.4 Implementation details

From Table 2.1 we read that for $N = 10^4$ and $\varepsilon = 10^{-6}$ we should use $p = 1.0654\ldots \cdot 10^{-6}$ as the parameter of Geo (p, L) distribution, utilized in reliable GeoGLE algorithm. Let us note, that we have to remember that in the framework of Algorithm 5, a draw is taken from Geo (p) distribution and further it is restricted to the interval $[0 : L]$. When $p \approx 10^{-6}$, the the generation of a number according to Geo (p) is simple. We may even use a trivial algorithm, which counts tosses of unfair coin before it falls heads up, where the probability of such the event in a single toss is p. To be more precise, we draw a number from Geo (p, L) distribution (with $L = 2^K - 1$), so once we toss a coin L times, then we reach the last possible value, hence the process stops. This schema is presented in Figure 2.7. The expected number of such the tosses is then $\mathrm{O}\left(\frac{1}{p}\right)$. Note that this method is not efficient as p is close to 0. Let us remark that usually efficient (in terms of runtime) pseudo-random number generators become inexact when p is very small.

When it comes to satisfy much stricter constraint for reliability of GeoGLE algorithm, a problem may arise. Indeed, for instance, if we choose $\varepsilon = 10^{-12}$ (and $N = 10^4$), then we should use parameter $p = 1.41009426\ldots \cdot 10^{-12}$. In that case a naive implementation of a random number generator with a geometric distribution Geo (p) may be burdened with significant numerical errors. To avoid this problem, we may use a clever method described in the paper of K. Bringmann and T. Friedrich [18] from ICALP'13. A main idea of this

method is to divide the draw of $G \sim \mathrm{Geo}\,(p)$ into two independent parts. First one is $a := G \div 2^k$, where $k \in \mathbb{N}$ is given as $\lfloor -\lg(p) \rfloor$. The second part is $b := G \bmod 2^k$. It may be not obvious at a first glance, but these two variables are, in fact, independent. In the first phase, we draw a number a according to $\mathrm{Geo}\left(1-(1-p)^{2^k}, \frac{L+1}{2^k} - 1\right)$. Note that $\frac{3}{4} \geqslant 1-(1-p)^{2^k} > 1 - e^{-\frac{1}{2}}$, so here we can use the standard method. The latter part is more technical. We repetitively draw $b \sim \mathrm{Uni}\left([0 : 2^k - 1]\right)$ until $B = 1$, where $B \sim \mathrm{Ber}\left((1-p)^b\right)$ is drawn after every redraw of b and the draws are independent. This clever trick utilize a rejection method in order to approximate a draw according to $\mathrm{Geo}\left(p, 2^k - 1\right)$. Note that when p is very close to 0, then probability mass functions of $\mathrm{Uni}\left([0 : 2^k - 1]\right)$ and $\mathrm{Geo}\left(p, 2^k - 1\right)$ are quite similar. The above rejection method is used in order to rectify the distortion and only needs a finite number of trials on average. Realize that then $a \cdot 2^k + b$ gives is stochastically equivalent to $\mathrm{Geo}\,(p, L)$ distribution. Additionally, we can decrease the implementation runtime. Indeed, b can be obtained lazily (bits of memory, where b is allocated are only determined when needed), since sometimes the most significant bits of b are enough to determine B. For more details, see [18].

2.8.5 Simplified solution

Let us also mention a simplified, approximated version of Theorem 2.8.1. It may be useful, when one has flexible approach to reliability and allows it to be slightly lower than $1 - \varepsilon$.

Theorem 2.8.2. *Suppose that* $0 < \varepsilon < \frac{2}{27}$ *and* $N \in \mathbb{N}\backslash\{1\}$. *Let*

1. $K = \left\lceil \lg\left(1 - \dfrac{W_{-1}\left(-\frac{2\varepsilon}{e^2(N-1)^2}\right)}{4\varepsilon}\right)\right\rceil$
2. $L = 2^K - 1$
3. $p = \dfrac{\ln\left((N-1)\sqrt{2L}\right)}{L}$

Then for every $n \in [N]$ *we have* $\Pr[S_{n,\mathrm{Geo}(p,L)}] \approx 1 - \varepsilon$.

As one can briefly see, first two formulas (for K and L) are very similar to theirs analogues from Theorem 2.8.1. The latter formula can be justified by Eq. (B.19) from Appendix B. Let us only note, that when ε is reasonably small, then parameters K and L are usually the same for Theorem 2.8.1 and Theorem 2.8.2. Moreover, parameters p are quite similar in both cases, especially when N is big and ε is small.[49]

[49] Then ε is a good approximation of $\tilde{\varepsilon}$ from Theorem 2.8.1, as one can see in Table B.1. Therefore the differences in the definitions of parameters are negligible as well in such the case.

2.9 A mixture of Geometric and Uniform LEA

2.9.1 Motivation

We have already provided that in the case of only two nodes (i.e. $n = 2$) the optimal probabilistic distribution (such that minimizes $\Pr[X = Y]$) on the set $[L]$ is the uniform distribution (see Section 2.5.4). In this case one may consider the following algorithm: nodes choose independently random numbers from the set $[0 : 2^K - 1]$ uniformly at random and the leader is the one which choose the biggest one. Nevertheless this approach is far from optimal (see Section 2.6).

We may also utilize a restricted version of geometric distribution instead of uniform distribution. Recall that properly used geometric random variables quickly reduce large groups of nodes to some small subsets.

However, it is possible to consider a mixture of both distributions: in the first phase we may use the restricted geometric distribution and the uniform one in the second phase. This solution was considered as a rectification of LGE algorithm in Section 2.7.2. Now, we are going to selected parameters properly, in order to improve slightly the reliability and sometimes spare some bits.

A description of GULE

Consider a quasi-anonymous Atlantic City $(1 - \varepsilon)$-reliable Leader Election algorithm. Assume that the procedure is divided in two phases, both compatible with Algorithm 2 or more precisely, it uses Algorithm 4 with some parameters L_1, p and $f(x) \equiv L_2 - 1$. The first phase is GeoGLE block, which is utilized to reduce the number of competitors for the leader title. All winners of the first part, continue the Election by executing the second phase, which is conducted according to ULE algorithm. We can assume, that all nodes take part in both elections and adapt the concept of the lexicographically ordered set of labels from Section 1.2.3 (just like in Section 2.7.3). Let N be the capacity of the network. We will only consider a case $n = N$, because of the monotonicity argument (see Theorem 2.9.1). We call this procedure a Geometric-Uniform Leader Election algorithm or shortly GULE algorithm.

Arbitrary simulations

In the first phase we concern urn model with $\mathrm{Geo}\left(p, 2^{K_1} - 1\right)$ distribution and another urn model with $\mathrm{Uni}\left(2^{K_2}\right)$ distribution for the second phase, for some $K_1, K_2 \in \mathbb{N}$, which will be obtained further (i.e. we assume $L_1 = 2^{K_1} - 1$ and $L_2 = 2^{K_2}$). Recall that $q = 1 - p$ ad let us denote $K = K_1 + K_2$.

Notice, that we have already considered the example of GULE algorithm with parameters $p = 0.01$, $K_1 = 14$, $K_2 = 80$, for network's capacity $N = 10^{26}$ and mistake frequency $\varepsilon = 10^{-23}$ (in Section 2.7.2). This approach assumes that $p = 0.01$ and find some sufficient parameters K_1 and K_2 and proves that 94 bits are enough. Can we do better, when p is not fixed at start?

Let us commence with a comparable example of performance of GULE algorithm.

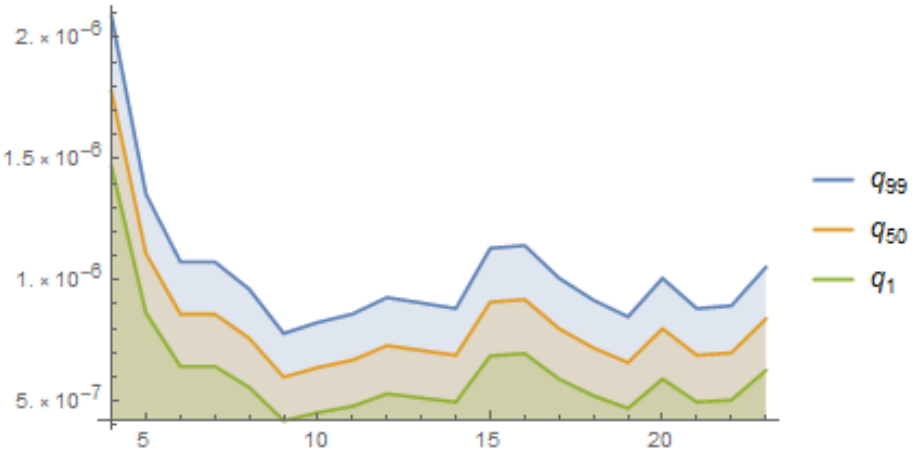

Figure 2.8: Plot of probabilities of failure (denoted by q_{50}) together with respective confidence intervals ($[q_1, q_{99}]$) at confidence level 0.98, simulated by Monte Carlo algorithms with 10^8 repetitions, according to GULE algorithm with $K = 23$ bits of memory in total, for $n = 100$ devices. A number of bits devoted to the first phase K_1 is given on abscissa. GeoGLE block is provided according to Geo $\left(p, 2^{K_1}\right)$ distribution, with $p = 0.1 \cdot 2^{7-K_1}$. The second phase of the algorithm utilize $23 - K_1$ bits in ULE block.

Figure 2.8 is a equivalent of Figure 2.6 for a mixture of GeoGLE and ULE algorithms with the same number of bits $K = 23$ in total and some arbitrary chosen parameters p, which depends on the number of rounds in the first phase of the algorithm, namely $p = 0.1 \cdot 2^{7-K_1}$. For GeoGLE algorithm, the most reliable parameter p occurred to be the one given by Theorem 2.8.1, which gave the average probability of failure for GeoGLE algorithm equals to $6.8 \cdot 10^{-7}$. In Figure 2.8, the better results were obtained for $K_1 \in \{9, 10, 11, 19\}$ and the best was $6 \cdot 10^{-7}$ (for $K_1 = 9$). Figure 2.8 affirms the efficiency of the mixture of GeoGLE and ULE blocks. Note that the parameter p was not even chosen very carefully. However, one can see, that there probably is some minimal number of bits needed to conduct the first phase in order to provide reliable LEA. Notice, that a case of $K_1 = 23$ is exactly GeoGLE algorithm. In this scenario, it seems, that for the assumed p, GeoGLE performs slightly worse than most of simulated GULE algorithms. Confidence intervals were provided again by de Moivre—Laplace Theorem 1.3.2, like for Figure 2.6.

2.9.2 Possible scenarios and monotonicity

In Table 2.2 we specify some possible events of GULE algorithm for n devices that we are going to discuss.

If one of the scenarios $\mathrm{Sc}_n{}^{(1a)}$, $\mathrm{Sc}_n{}^{(1b)}$ holds, then we shortly denote it as a

i	Scenario $\mathrm{Sc}_n{}^{(i)}$	$\Pr\left[\mathrm{Sc}_n{}^{(i)}\right]$
1a.	Successful GeoGLE block	α
1b.	Unsuccessful GeoGLE block, rectified by ULE one	β
2.	Two or more devices draws $2^{K_1}-1$ in GeoGLE phase	$< \varepsilon_1$
3.	Both GeoGLE and ULE blocks fail, when at most one node drawn $2^{K_1}-1$ in the first phase	$< \varepsilon_2$

Table 2.2: Possible scenarios of a single run of GULE algorithm and theirs parametrized probabilities.

scenario $\mathrm{Sc}_n{}^{(1)}$, which entails a successful Leader Election (with an additional unnecessary condition, that there is no collision in L_1-th urn in the first phase). Briefly, $\mathrm{Sc}_n{}^{(3)}$ must not end with the choice of the unique leader and $\mathrm{Sc}_n{}^{(2)}$ may be eventually resolved in ULE phase. Therefore, we assume, that $\alpha+\beta > 1-\varepsilon$ and $\varepsilon_1+\varepsilon_2 \leqslant \varepsilon$, in order to provide $(1-\varepsilon)$-reliable LEA.

Theorem 2.9.1 (Monotonicity of probabilities of successful GULE algorithm)**.** *Let $\varepsilon > 0$, $N \in \mathbb{N}\backslash\{1\}$ and $\mathrm{Sc}_n{}^{(1)}[p, K_1, K_2]$ denotes the successful GULE algorithm for parameters p, K_1 and K_2. If $\Pr\left[\mathrm{Sc}_N{}^{(1)}[p, K_1, K_2]\right] > 1-\varepsilon$, then*

$$(\forall\, n < N)\ 1-\varepsilon < \Pr\left[\mathrm{Sc}_N{}^{(1)}[p, K_1, K_2]\right] < \Pr\left[\mathrm{Sc}_n{}^{(1)}[p, K_1, K_2]\right] \ .$$

Proof. Fix p, K_1, K_2 and let $L_i = 2^{K_i}-1$, for $i \in [2]$. At first realize, that then $\Pr\left[\mathrm{Sc}_N{}^{(2)}\right] > \Pr\left[\mathrm{Sc}_{N-1}{}^{(2)}\right]$. Indeed, the 2. scenario occurs when no more than 1 device draws L_1, so $\Pr\left[\mathrm{Sc}_N{}^{(2)}\right] = 1-\left(1-q^{L_1}\right)^N - Nq^{L_1}\left(1-q^{L_1}\right)^{N-1}$. Hence

$$\begin{aligned}\Pr\left[\mathrm{Sc}_N{}^{(2)}\right] - \Pr\left[\mathrm{Sc}_{N-1}{}^{(2)}\right] &= q^{L_1}\left(1-q^{L_1}\right)^{N-1} + (N-1)\left(q^{L_1}\right)^2\left(1-q^{L_1}\right)^{N-2} \\ &\quad - q^{L_1}\left(1-q^{L_1}\right)^{N-1} \\ &= (N-1)\left(q^{L_1}\right)^2\left(1-q^{L}\right)^{N-2} > 0\ .\end{aligned}$$

By induction, we obtain $\Pr\left[\mathrm{Sc}_N{}^{(2)}\right] > \Pr\left[\mathrm{Sc}_n{}^{(2)}\right]$ for $n < N$.
The 3. scenario is more complex. Consider that $(T_i)_{i=1}^N$ are i.i.d. random variables with $\mathrm{Geo}\,(p, L_1)$ distribution. Let $[t_i]_{i=1}^N$ be a realization of a random vector $[T_i]_{i=1}^N$. We want to compare probabilities of some events using tuples $(t_1, \ldots, t_N)$ and $(t_1, \ldots, t_{N-1})$. Let us denote a situation when the first tuple elects a leader correctly and the second one falls into $\mathrm{Sc}_{N-1}{}^{(3)}$ by A_N, and a situation when the second tuple elects a leader correctly and the first one falls into $\mathrm{Sc}_N{}^{(3)}$ by B_N. Let $m := \max(t_1, \ldots, t_{n-1})$. A_N means that $t_N > m$ and there are at least two indices $i \in [n-2]$, $j \in [i+1 : n-1]$ such that $t_i = t_j = m$, for which ULE phase does not resolve the collision, hence $\Pr[A_n] <$

$q^{m+1}\Pr[W_{\geqslant 2,n-1,p}]$. B_N occurs, for instance, when there was the unique maximum in the set $\{t_1,\ldots,t_{n-1}\}$ and $t_n = m$, hence $\Pr[B_n] > pq^m\Pr[W_{1,n-1,p}]$. Realize that Theorem 1.3.3 and $p < \frac{1}{2}$ gives

$$\begin{aligned}\frac{\Pr[B_n]-\Pr[A_n]}{q^m} &> p\Pr[W_{1,n-1,p}] - (1-p)(1-\Pr[W_{1,n-1,p}]) \\ &= \Pr[W_{1,n-1,p}] + p - 1 > p - 1 - \frac{p}{\ln(1-p)} + \frac{p}{3}\ln(1-p) \overset{p<\frac{1}{2}}{>} 0\ .\end{aligned}$$

Therefore we obtained $\Pr\left[\mathrm{Sc}_N{}^{(3)}\right] > \Pr\left[\mathrm{Sc}_{N-1}{}^{(3)}\right]$. Again, a similar induction ends the considerations for the 3. scenario. Finally, after putting together the results for both problematic scenarios we receive the thesis. □

Theorem 2.9.1 let us consider only a non-anonymous GULE with N devices, since it will be appropriate for all $n < N$ as well.

2.9.3 Maximum of Geometric distributions

Definition 2.9.1. *Let $t \in \mathbb{N}$ and $(G_i)_{i=1}^n$ be a sequence of i.i.d random variables with distribution* $\mathrm{Geo}(p)$. *We define* $\mathrm{MG}_{\geqslant t}(n) := \mathrm{card}\left(\{i \in [n] : G_i \geqslant t\}\right)$, *i.e. how many of n realizations of independent geometric variables are at least t.*

Proposition 1. *Let $t \in \mathbb{N}$. Then*

$$\Pr[\mathrm{MG}_{\geqslant t}(n) < 2] = \left(1 + (n-1)q^t\right)\left(1-q^t\right)^{n-1}\ .$$

Proof. For $k \in [0:n]$ we have

$$\Pr[\mathrm{MG}_{\geqslant t}(n) = k] = \binom{n}{k}\Pr[G_1 \geqslant t]^k \Pr[G_1 < t]^{n-k} = \binom{n}{k}\left(q^t\right)^k\left(1-q^t\right)^{n-k}, \tag{2.21}$$

therefore, in particular

$$\Pr[\mathrm{MG}_{\geqslant t}(n) < 2] = \left(1-q^t\right)^n + nq^t\left(1-q^t\right)^{n-1} = \left(1+(n-1)q^t\right)\left(1-q^t\right)^{n-1}.$$

□

Corollary 1. *Let $L \in \mathbb{N}$ and $\lambda = (n-1)q^L$. Then*

$$\Pr[\mathrm{MG}_{\geqslant L}(n) < 2] = (1+\lambda)\left(1 - \frac{\lambda}{n-1}\right)^{n-1}\ .$$

Proposition 2. *Let $L \in \mathbb{N}$ and $\lambda = (n-1)q^L < 1$. Then*

$$\lambda^2\frac{1-\lambda}{2} < \Pr[\mathrm{MG}_{\geqslant L}(n) \geqslant 2] < \lambda^2\ .$$

Proof. From the definition, $\lambda \in (0, n-1)$, so from Corollary 1 and Weierstrass' Product Inequality (Theorem 1.4.2) we conclude that

$$\Pr[\mathrm{MG}_{\geqslant L}(n) \geqslant 2] = 1 - (1+\lambda)\left(1 - \frac{\lambda}{n-1}\right)^{n-1} \tag{2.22}$$

$$\overset{\text{WPI}}{\geqslant} 1 - (1+\lambda)\left(1 - \lambda + \frac{(n-2)\lambda^2}{2(n-1)}\right)$$

$$= \lambda^2\left(1 - \frac{(n-2)(1+\lambda)}{2(n-1)}\right) > \lambda^2\frac{1-\lambda}{2}\ . \tag{2.23}$$

Again, from (2.22) and Weierstrass' Product Inequality (Theorem 1.4.2) we get $\Pr[\mathrm{MG}_{\geqslant L}(n) \geqslant 2] < 1 - (1+\lambda)(1-\lambda) = \lambda^2$. □

Note that ${\mathrm{Sc}_n}^{(2)}$ is stochastically equivalent with $(\mathrm{MG}_{\geqslant L}(n) \geqslant 2)$.

Let us announce that part of the proof of Proposition 2 will be utilized in the proof of Theorem 2.8.1 in Appendix B.

2.9.4 Limitations on failures probabilities

Lemma 9. *Assume that* $\lambda = (n-1)q^{L_1} < 1$ *and*

$$-L_1 \ln(1-p) > \ln(n) - \frac{1}{2}\ln(\varepsilon_1)\ .$$

Then the condition $\Pr\left[{\mathrm{Sc}_n}^{(2)}\right] < \varepsilon_1$ *is satisfied.*

Proof. From Proposition 2, $\lambda^2 < \varepsilon_1$ is sufficient to provide $\Pr\left[{\mathrm{Sc}_n}^{(2)}\right] < \varepsilon_1$. However that entails $q^{L_1} < \frac{\sqrt{\varepsilon_1}}{n-1}$ or equivalently $-L_1\ln(1-p) > \ln(n-1) - \frac{1}{2}\ln(\varepsilon_1)$. □

Lemma 10. *Let* $p < \frac{1}{2}$. *Then* $\Pr\left[{\mathrm{Sc}_n}^{(3)}\right] < \frac{p}{L_2(1-p)^2}$ *for any* $n \in \mathbb{N}$.

Proof. By $F_{a,\mathrm{Uni}(L_2)}$ we denote a failure in ULE block of algorithm for a devices, i.e. that there is not a unique maximum amongst a i.i.d. realizations of variables with $\mathrm{Uni}\,(L_2)$ distribution. From Theorem 1.3.3, we know that $\Pr[W_{a,n,p}] < -\frac{p^a}{a\ln(1-p)} - \frac{(a+1)^2p^a\ln(1-p)}{12a}$ for any $n \in \mathbb{N}$. Theorem 2.6.2 bears an inequality $\Pr\left[F_{a,\mathrm{Uni}(L_2)}\right] < \frac{a}{2L_2}$, so from a fact that $\ln 2 > -\ln(1-p) > p$, we provide the following:

$$\sum_{a=2}^{n}\Pr[W_{a,n,p}]\Pr\left[F_{a,\mathrm{Uni}(L_2)}\right] < \sum_{a=2}^{n}\left(-\frac{p^a}{a\ln(1-p)} - \frac{(a+1)^2p^a\ln(1-p)}{12a}\right)\frac{a}{2L_2}$$

$$< \frac{1}{2L_2}\sum_{a=2}^{n}\left(p^{a-1} + \frac{(a+1)^2p^a\ln 2}{12}\right)$$

In Appendix C we will prove Lemma 22, which shows that:

$$\sum_{a=2}^{n}\left(p^{a-1}+\frac{(a+1)^2p^a\ln 2}{12}\right)<\frac{p}{(1-p)^2}\ .$$

It remains to realize to realize that, if there is a collision in Leader Election algorithm in urn model, according to $\mathrm{Geo}\,(p)$ (without restriction), and the maximum amongst identifiers is less than L_1, then in its restricted version, according to $\mathrm{Geo}\,(p, L_1)$ does not affect any of the identities, so

$$\Pr\left[\mathrm{Sc}_n{}^{(3)}\right]<\sum_{a=2}^{n}\Pr[W_{a,n,p}]\Pr\left[F_{a,\mathrm{Uni}(L_2)}\right]\ .$$

□

2.9.5 Derivation of parameters

Minimizing the number of rounds in both phases

For a concise path of determining the parameters we combine the results from Lemma 9:

$$L_1+1>-\frac{\ln(n-1)-\frac{1}{2}\ln(\varepsilon_1)-\ln(1-p)}{\ln(1-p)}>\frac{\ln(n-1)-\frac{1}{2}\ln(\varepsilon_1)}{p} \tag{2.24}$$

with the one from Lemma 10:

$$L_2>\frac{p}{2(1-p)^2\varepsilon_2}\ . \tag{2.25}$$

When we change the notation via substitutions $L_1=2^{K_1}-1$ and $L_2=2^{K_2}$, we obtain

$$2^{K_1+K_2}>\frac{\ln(n-1)-\frac{1}{2}\ln(\varepsilon_1)}{2(1-p)^2\varepsilon_2}\ . \tag{2.26}$$

We show further that ε_1 and ε_2 can be derived from n and ε, so we may assume that all the variables on the right side of the above inequality are provided a priori. We desire to optimize the time complexity, i.e. K_1+K_2 rounds. From the formula (2.26) we find its minimal possible value:

$$\mu(K_1+K_2):=\left\lceil\lg\left(\frac{\ln(n-1)-\frac{1}{2}\ln(\varepsilon_1)}{\varepsilon_2}\right)\right\rceil-1\ . \tag{2.27}$$

We assumed that $p<\frac{1}{2}$, so $(1-p)^2$ varies in a set $(\frac{1}{4},1)$. It is necessary to convince that we can reach threshold from (2.27), but from (2.26) we can easily see, that it is only possible when $(1-p)^2>\frac{1}{2}$, so we can assume without a loss of generality that $p<1-\frac{1}{\sqrt{2}}$.

Analogically, from inequalities (2.24) and (2.25) we deduce that

$$K_1\geqslant\left\lceil\lg\left(\ln(n-1)-\frac{1}{2}\ln(\varepsilon_1)\right)-\lg p\right\rceil$$

and

$$K_2 \geqslant \left\lceil -\lg(\varepsilon_2) - 1 + \lg\left(\frac{p}{(1-p)^2}\right)\right\rceil \ .$$

Let us therefore define

$$\begin{cases} e_1(p) := \lg\left(\ln(n-1) - \frac{1}{2}\ln(\varepsilon_1)\right) - \lg p \\ e_2(p) := -\lg(\varepsilon_2) - 1 + \lg\left(\frac{p}{(1-p)^2}\right) \end{cases}$$

and $m_j(p) := \lceil e_j(p) \rceil$ for $j \in [2]$. Realize that $\lceil a \rceil + \lceil b \rceil \geqslant \lceil a+b \rceil$, so

$$\begin{aligned} m_1(p) + m_2(p) &\geqslant \left\lceil \lg\left(\ln(n-1) - \frac{1}{2}\ln(\varepsilon_1)\right) - \lg(\varepsilon_2) - 1 - 2\lg(1-p)\right\rceil \\ &\geqslant \left\lceil \lg\left(\frac{\ln(n-1) - \frac{1}{2}\ln(\varepsilon_1)}{\varepsilon_2}\right)\right\rceil - 1 = \mu(K_1+K_2) \end{aligned}$$

and if the number of used bits is indeed $\mu(K_1+K_2)$ then the equalities must hold. Note that $\lceil a \rceil + \lceil b \rceil = \lceil a+b \rceil$, when $\{a\}_f + \{b\}_f \geqslant 1$.[50] Since we want to minimize $m_1(p)+m_2(p)$, we are interested in such p, that $\{e_1(p)\}_f + \{e_2(p)\}_f \geqslant 1$ and

$$-2\lg(1-p) \leqslant 1 - \left\{\lg\left(\frac{\ln(n-1) - \frac{1}{2}\ln(\varepsilon_1)}{\varepsilon_2}\right)\right\}_f \ . \tag{2.28}$$

Let say that p_0 is the biggest p, which satisfies (2.28).

We are going to consider the aforementioned expressions with abandoned fractional parts in order to optimize $\mu(K_1+K_2)$, depending on a priori parameters n and ε. It was mentioned that we will establish ε_1 and ε_2, basing only on n and ε. Here comes the time to reveal the secret. One may choose $\frac{\varepsilon}{2}$ for both parameters, but we prefer to do this more optimally, because the arbitrary apportionment can not guarantee the minimal time complexity. We naturally assume that $\varepsilon_1 + \varepsilon_2 = \varepsilon$ and we are going to utilize Lagrange multipliers (this method is described e.g.[13]). Let us define a function:

$$h(\varepsilon_1, \varepsilon_2) := \frac{\ln(n-1) - \frac{1}{2}\ln(\varepsilon_1)}{\varepsilon_2} + (\varepsilon_1 + \varepsilon_2 - \varepsilon)\lambda \ .^{51}$$

Then $\frac{\partial h(\varepsilon_1,\varepsilon_2)}{\partial \varepsilon_1} = -\frac{1}{2\varepsilon_1\varepsilon_2} + \lambda$ and $\frac{\partial h(\varepsilon_1,\varepsilon_2)}{\partial \varepsilon_2} = -\frac{\ln(n-1)-\frac{1}{2}\ln(\varepsilon_1)}{\varepsilon_2^2} + \lambda$. Moreover, if $\varepsilon < 0.6$, then a Hessian matrix of h is positively defined for any pair $(\varepsilon_1, \varepsilon_2)$ from the interval $\{(x, \varepsilon - x) : x \in (0, \varepsilon)\}$:[52]

$$H = \begin{bmatrix} \dfrac{1}{2\varepsilon_1^2\varepsilon_2} & \dfrac{1}{2\varepsilon_1\varepsilon_2^2} \\ \dfrac{1}{2\varepsilon_1\varepsilon_2^2} & \dfrac{2\ln(n-1) - \ln(\varepsilon_1)}{\varepsilon_2^3} \end{bmatrix}$$

[50] Recall that $\{x\}_f$ is a fractional part of x.

[51] It should be clear that $h(\varepsilon_1, \varepsilon_2) = m_1(p) + m_2(p)$.

[52] Roughly speaking, a Hessian of a function f, which has continuous all second partial derivatives with respect to all k variables, is a square matrix $\left[\frac{\partial^2}{\partial_i \partial_j} f\right]_{k \times k}$.

(since $4\ln(n-1) - 2\ln(\varepsilon_1) - 1 > 0$ whenever $n > 2$ or $\varepsilon_1 < e^{-\frac{1}{2}} = 0.6065\ldots$). Hence h reaches minimum when $\frac{1}{2\varepsilon_1\varepsilon_2} = \frac{\ln(n-1)-\frac{1}{2}\ln(\varepsilon_1)}{\varepsilon_2^2}$ or alternatively $\varepsilon_2 = \varepsilon_1\left(2\ln(n-1) - \ln(\varepsilon_1)\right)$, what provides

$$\varepsilon = \varepsilon_1\left(1 + 2\ln(n-1) - \ln(\varepsilon_1)\right) = \varepsilon_1 \ln\left(\frac{(n-1)^2 e}{\varepsilon_1}\right) .$$

We easily see that $\dfrac{-\varepsilon}{(n-1)^2 e} = \dfrac{\varepsilon_1}{(n-1)^2 e}\ln\left(\dfrac{\varepsilon_1}{(n-1)^2 e}\right)$, what can be translated to

$$\exp\left(W\left(\frac{-\varepsilon}{(n-1)^2 e}\right)\right) = \frac{\varepsilon_1}{(n-1)^2 e} .$$

Multiplication by $W\left(\frac{-\varepsilon}{(n-1)^2 e}\right)$ shows that $\dfrac{-\varepsilon}{(n-1)^2 e} = \dfrac{\varepsilon_1 W\left(\frac{-\varepsilon}{(n-1)^2 e}\right)}{(n-1)^2 e}$, what finally results in:

$$\varepsilon_1 = -\frac{\varepsilon}{W\left(\dfrac{-\varepsilon}{(n-1)^2 e}\right)} . \tag{2.29}$$

The argument of the above W-Lambert (multi-)function is negative, so two real values are adaptable for a result. Quick look shows that we have to take W_{-1} branch (see e.g. Figure 1.1). If we assume a priori the value of ε, then we can attain an optimal ε_1 given by above formula (2.29) and put $\varepsilon_2 = \varepsilon - \varepsilon_1$.

After obtaining ε_1, we can struggle easily with m_1. Since $-\lg(p)$ is decreasing function and $p \leqslant p_0$, we realize that

$$m_1(p) \geqslant \left\lceil \lg\left(\ln(n-1) - \frac{1}{2}\ln(\varepsilon_1)\right) - \lg p_0 \right\rceil .$$

Let assume for a while that we prefer to minimize $m_1(p)$ rather than $m_2(p)$. In this situation, the above inequality turns into equality. This way a new boundary for p is at our fingertips, namely:

$$-\lg(p) + \lg(p_0) = e_1(p) - e_1(p_0) \leqslant 1 - \left\{\lg\left(\ln n - \frac{1}{2}\ln(\varepsilon_1)\right) - \lg(p_0)\right\}_f .$$

Let us define p_1 in such a way that $e_1(p_1) = m_1(p_0)$, what bears $p_1 < p < p_0$.

Observation 1. *Note that, if we do not want to minimize $m_1(p)$, we may define $p_k := \frac{p_1}{2^{k-1}}$ and consider $p \in (p_k, p_{k-1})$, which provides $m(p) = m(p_0) + k$.*

With the above capture, we may now deal with $m_2(p)$. It is easy to obvious that $\lg\left(\frac{p}{(1-p)^2}\right)$ is ascending and negative function for $p < 1 - \frac{1}{\sqrt{2}}$, so depending on previously chosen k,[53]

$$m_2(p) = \left\lceil -\lg(\varepsilon_2) - 1 + \lg\left(\frac{p_k}{(1-p_k)^2}\right)\right\rceil$$

[53] A range of admissible k will be provided later.

Remark that now $m_1(p)$ is fixed and we are not able to increase $m_2(p)$, so the inequality has been displaced by the equality. We also achieved an additional constraint for p, namely:

$$\lg\left(\frac{p}{(1-p)^2}\right) - \lg\left(\frac{p_k}{(1-p_k)^2}\right) \leqslant 1 - \left\{-\lg(\varepsilon_2) - 1 + \lg\left(\frac{p_k}{(1-p_k)^2}\right)\right\}_f \ .$$

Let us define p'_k in such the way that $e_2(p'_k) = m_2(p_k)$. This way we reach $p_k < p < \min(p_{k-1}, p'_k)$.

Proposition 3. $p'_k < 2p_k$.

Proof. From definition of p'_k, we know that $\lg\left(\frac{p'_k}{(1-p'_k)^2}\right) - \lg\left(\frac{p_k}{(1-p_k)^2}\right) < 1$ or equivalently $\frac{p'_k}{(1-p'_k)^2} < 2\frac{p_k}{(1-p_k)^2}$. But $p_k < p'_k$, so $(1-p_k)^2 > (1-p'_k)^2$. It proves that $p'_k < 2p_k$. □

Remark Realize that for any appropriate $k \neq 1$, $p'_k < 2p_k = p_{k-1}$. From the definition of p_1, it should also be clear that $p_0 < 2p_1$.

For now, we established some limitations on p, $m_1(p)$ and $m_2(p)$. Let us return to the condition $\{e_1(p)\}_f + \{e_2(p)\}_f \geqslant 1$ mentioned at the beginning:

Theorem 2.9.2. *If* $p \in (p_k, \min(p_{k-1}, p'_k))$, *then* $\{e_1(p)\}_f + \{e_2(p)\}_f \geqslant 1$.

Proof. Realize that $\{e_1(p_k)\}_f = 0$ and $\{e_2(p'_k)\}_f = 0$ from the definitions of p_k and p'_k respectively. By Proposition 3 and definitions of p_k and p'_k, we also know that $\lfloor e_2(p) \rfloor = \lfloor e_2(p_k) \rfloor$ for $p \in (p_k, p'_k)$. Notice that $\frac{\partial}{\partial p} e_1(p) = -\frac{1}{p} < 0$ and $\frac{\partial}{\partial p} e_2(p) = \frac{1}{p} + \frac{2}{1-p} > 0$, so $\frac{\partial}{\partial p}(e_1(p) + e_2(p)) = \frac{2}{1-p} > 0$. This shows that $\{e_1(p)\}_f$ is continuous and descending for $p \in (p_k, p'_k)$ and by Proposition 3 $\{e_2(p)\}_f$ is continuous and ascending on $[p_k, p'_k)$. Since $e_1(p)$ is continuous and decreasing for $p > 0$, and $\{e_1(p_k)\}_f = 0$, we obtain that

$$\lim_{p \to p_k^+} \{e_1(p)\}_f = 1 \ . \tag{2.30}$$

From Mean Value Theorem 1.2.1 we obtain

$$e_1(p) + e_2(p) = e_1(p_k) + e_2(p_k) + \frac{2(p - p_k)}{1 - p^*} \ ,$$

for $p \in (p_k, p'_k)$ and some $p^* \in (p_k, p)$. Since $\frac{1}{1-p^*} > \frac{1}{1-p_k}$ and (2.30), we attain

$$\{e_1(p)\}_f + \{e_2(p)\}_f > 1 + \{e_2(p_k)\}_f + \frac{2(p - p_k)}{1 - p_k} > 1$$

for $p \in (p_k, p'_k)$. By Proposition 3, the original statement is also true. □

Theorem 2.9.2, together with the previous argument, clarify a manner of choice of all parameters except k and p.

Motivation of choice of k and p parameters

Let us realize that $\mathrm{Sc}_n{}^{(2)}$ does not need to imply a failure of the GULE algorithm, contrary to $\mathrm{Sc}_n{}^{(3)}$. Therefore we want to tune up the parameter k in such the way that it maximizes the probability that ULE block of algorithm will resolve the collision from the first phase. Note that, when we increase K_2 by 1, then:

- K_1 decreases by 1,
- p parameter increases approximately twice,
- λ is quite similar, because $\lambda = (n-1)(1-p)^{2^{K_1}-1} \approx (n-1)(1-2p)^{2^{K_1-1}-1}$,
- A number of devices, which draws $2^{K_1}-1$-th urn in the first phase is closely the same. Indeed, from (2.21) and a constraint $\lambda < \sqrt{\varepsilon}$, we get

$$\Pr[\mathrm{MG}_{\geqslant L}(n) = a] = \frac{[n]_a}{a!} q^{La} \left(1 - \frac{\lambda}{n-1}\right)^{n-k} \approx \frac{\lambda^a}{a!} e^{-\lambda} ,$$

  which does not change significantly,
- $\Pr\left[F_{a,\mathrm{Uni}\left(2^{K_2}\right)}\right]$ decreases approximately twice for the same number of competing devices in the second phase (see Theorem 2.6.2).

Therefore we suggest to choose the biggest possible K_2, which is related to $k = 1$.[54]

Let us notice that parameter p does not need to be chosen as precise as in Theorem 2.8.1, since, the second phase of GULE algorithm resolves collisions and p is mainly used to reduce the initial number of stations. However, according to Theorem 1.3.3, the probability of collisions slightly increases with respect to p, so we should be likely to choose possibly small parameter $p \in (p_1, p_0)$. Nevertheless, we cannot simply use p_1, hence we may, for instance, put $p = 0.99p_1 + 0.01p_0$.

2.9.6 Main contribution

The beneath theorem summarize all the previous arguments gathered in this section:

Theorem 2.9.3. *Let N be a capacity of the network and $\varepsilon < 0.6$. Moreover, let us define:*

1. $\varepsilon_1 = -\dfrac{\varepsilon}{W_{-1}\left(\dfrac{-\varepsilon}{(N-1)^2 e}\right)}$,
2. $\varepsilon_2 = \varepsilon - \varepsilon_1$,

[54] Let us mention, that at some point, some of the above approximations may begin to be inaccurate, hence the biggest possible K_2 does not have to be the best solution. Nevertheless, the previous argument guarantees the appropriate level of reliability.

3. $p_0 = 1 - 2^{-\frac{\ell_0}{2}}$, *where:*

$$\ell_0 = -2\lg(1-p_0) = 1 - \left\{\lg\left(\ln(N-1) - \frac{1}{2}\ln(\varepsilon_1)\right) - \lg(\varepsilon_2)\right\}_f ,$$

4.

$$K_1 = \left\lceil \lg\left(\ln(N-1) - \frac{1}{2}\ln(\varepsilon_1)\right) - \lg(p_0)\right\rceil$$

5. $p_1 = p_0 2^{-\ell_1}$, *where:*

$$\ell_1 = -\lg\left(\frac{p_1}{p_0}\right) = 1 - \left\{\lg\left(\ln(N-1) - \frac{1}{2}\ln(\varepsilon_1)\right) - \lg(p_0)\right\}_f .$$

6.

$$K_2 = \left\lceil -\lg(\varepsilon_2) - 1 + \lg\left(\frac{p_0}{1-p_0}\right)\right\rceil$$

7. $L_1 = 2^{K_1} - 1$, $L_2 = 2^{K_2}$, $p = 0.99p_1 + 0.01p_0$.

Let $S_{n,\mathrm{GULE}(p,L_1,L_2)}$ *denotes a successful GULE algorithm with the first phase according to* $\mathrm{Geo}\,(p, L_1)$ *distribution and the second one, with* $\mathrm{Uni}\,(L_2)$ *distribution. Then*

$$(\forall\, n \leqslant N)\ \Pr\left[S_{n,\mathrm{GULE}(p,L_1,L_2)}\right] > 1 - \varepsilon\ .$$

Let us remark, that according to Section 2.9.5, one can modify some of the parameters provided in Theorem 2.9.3.

2.9.7 Comparisons

Let us consider two examples which was mentioned at the very beginning of Section 2.9.

The first was analyzed in Section 2.7.2, with initial parameters $N = 10^{26}$ and $\varepsilon = 10^{-23}$, where we have obtained $p = 0.01$, $K_1 = 14$ and $K_2 = 80$, so $K = 94$ rounds in total. Theorem 2.9.3 provides $p = 0.0218802\ldots$, $K_1 = 12$ and $K_2 = 71$, hence $K = 83$ in total. According to our argument in Section 2.9.5, one can alternatively choose $p = 0.01094\ldots$, $K_1 = 13$ and $K_2 = 70$, what shows that in Section 2.7.2 the parameters were provided rather inefficiently. Let us note, that in this scenario, Theorem 2.8.1 provides even better $K = 82$, however with $p = 1.832883\ldots \cdot 10^{-23}$, which is theoretically better than GULE, however it this case, evaluation can be easily burdened with significant numerical errors.

Second example is related to Figure 2.8, which was plotted for $n = 100$ and $\varepsilon = 10^{-6}$. Theorem 2.9.3 provides $p = 0.0518567\ldots$, $K_1 = 8$ and $K_2 = 16$, what adds up to $K = 24$. On the other hand, Theorem 2.8.1 gives $K = 23$ and $p = 1.53932\ldots 10^{-6} = 0.100881 \cdot 2^{-16}$ in the same scenario. Thence, GeoGLE solution is theoretically better. However, let us note, that in Figure 2.8 we

also included a case of GeoGLE algorithm[55] with parameters $K = 23$ and $p = 0.1 \cdot 10^{-16}$, which are very close to these from Theorem 2.8.1. As we can easily see in Figure 2.8, GULE algorithm still can perform better than GeoGLE algorithm in similar arrangement. This remark sheds a new light on GULE solution. Indeed, in the optimization of the total number of bits that have to be use for reliable algorithm, we have omitted a case in ${\mathrm{Sc}_n}^{(2)}$, when ULE block resolves collisions. We postpone the rectification of GULE algorithm to the future work. It is not excluded, that rectified version can use even less bits than GeoGLE algorithm.

2.10 Summary, comparisons and future work

At first, in Section 2.5 we attained optimal distributions for Leader Election algorithms in urn model, in non-anonymous arrangement (in terms of reliability, for a given support). In Section 2.5.6 we showed that these optimal solutions always utilize either $-\lceil \ln(\varepsilon) \rceil + 1$ or $-\lceil \ln(\varepsilon) \rceil$ bits of memory, independent of the given network's size, where ε is the mistake frequency. In Appendix A we provide also efficient approximations of these optimal distributions, which can be obtained much faster than the optimal ones.

We showed that two classical Leader Election algorithms from [89] and [64] are based on the same framework, described in Section 2.3, whose idea consists in choosing the node with the largest randomly generated identifier. We proposed Uniform Leader Election (ULE) algorithm in Section 2.6 and showed that it is a restricted version of procedure from [89]. We provided a monotonicity property of ULE with respect to the number of nodes. Such the property of Leader Election algorithms is often not easy to prove explicitly and usually it is even not satisfied (as we have showed e.g. for Geometric Green Leader Election (GeoGLE) algorithm in Section B.4).

In Section 2.8 we proposed an alternative adaptation of Green Leader Election algorithm from [64], namely GeoGLE algorithm with very precisely set parameters. We justified why the parameters of this LEA should be established very carefully. We also presented in Section 2.8.4 some implementation details inextricably linked with drawing numbers according to geometric distribution with small parameter p, basing on a known solution.

We showed that Splitting and Naming procedures from [79] can be adapted to urn model and we provided several results concerning probabilities of success of these algorithms for small number of devices, which questions the utility of this LEAs in such the scenario.

In Section 2.9 we provided GULE algorithm, which utilizes the rectified version of Splitting and Naming framework (Algorithm 4) and we showed that this arrangement is comparable (in terms of time complexity) with GeoGLE algorithm and that GULE solution can be still improved. Nevertheless GULE solution can be still utilized instead of GeoGLE one, when p parameter in the

[55] It is a GULE algorithm with $K_2 = 0$.

second arrangement is unimaginably small, since the execution of GeoGLE solution may be then burdened with significant numerical errors, what can impact the reliability.

We mainly focused on the number of bits needed to perform a $(1-\varepsilon)$-reliable Leader Election. Therefore we analyzed this property in all arrangements aforementioned above and we provided several examples of rectifications of some of these algorithms. We showed, that in the regime of quasi-anonymous networks, our solution (GeoGLE algorithm) is the best amongst the considered in this dissertation, and to the best of our knowledge, there are not any others, which are competitive except GULE presented in Section 2.9.

In order to summarize the results of quasi-anonymous Leader Election algorithms, let us compare almost all of the aforementioned procedures in similar conditions (we omit GULE algorithm, since we know that it is not well optimized), i.e. we restrict the memory reserved to save identifiers to $K = 15$ bits and assume the network's capacity to be $N = 500$. Results are presented in Figure 2.9. Splitting and Naming procedures had to be divided in two restricted parts. In order to accomplish it effectively, we utilize at most $K_1 = 3$ bits in S-N w.h.p. and $K_1 = 4$ bits in S-N w.v.h.p. for the first blocks of algorithms. The second phases were provided with $\mathrm{Uni}\left(\max\left(f(t_v), 2^{15-K_1}\right)\right)$ in order to reduce the collisions (t_v is the maximal identifier from the first phase and f is determined by particular algorithm.

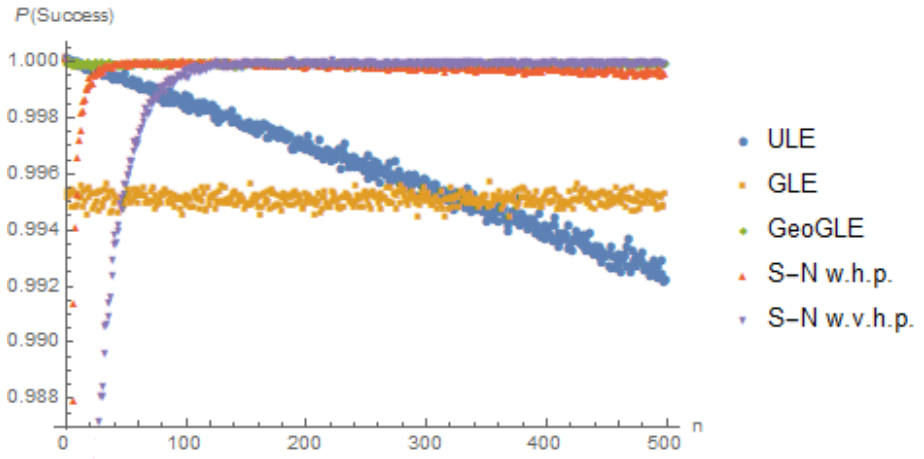

Figure 2.9: Comparison of simulated probabilities of success of five Atlantic City versions of Splitting and Naming algorithms restricted to 15 bits of memory. We conducted 10^5 experiments for each size of the network $n \in [500]$ and each of the algorithms.

One can easily see, that both LGE and GeoGLE algorithms keep almost constant reliability for all $n \in [500]$, ULE one is linearly dropping the reliability. Splitting and Naming algorithms together with GeoGLE one performs definitely

better than the latter two, so in Figure 2.10 we consider only the best 3, with the higher precision and wider interval. One can briefly realize that Splitting and Naming solutions performs terribly for $n < 50$ and $n < 150$, respectively in w.h.p and w.v.h.p. versions of protocol. Moreover, S-N w.h.p. outperforms the other two for $n \in [50, 100]$, however it further looses the reliability approximately linearly. However, for $n > 150$ both rectified version of S-N w.v.h.p. and GeoGLE operate comparably. Indeed, we have checked that the mean of simulated reliabilities for $n > 150$ are respectively 0.999824 and 0.99981, however, for all $n \in [750]$, the means change to 0.998002 and 0.999811. This affirms that GeoGLE algorithm keeps the reliability on the same level, which one cannot say about Splitting and Naming algorithms.

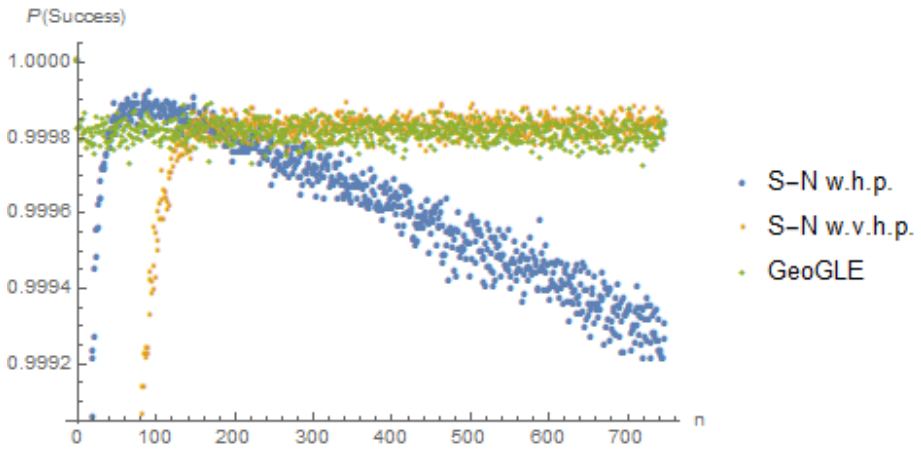

Figure 2.10: Comparison of simulated probabilities of success of two Atlantic City versions of Splitting and Naming algorithms restricted to 15 bits of memory. We conducted $3 \cdot 10^5$ experiments for each size of the network $n \in [750]$ and each of the algorithms.

Future work

A quality of approximations of optimal NALEAs from Appendix A was not considered in the terms of the number of bits. We would like to consider it in the future, as well as the improvement of GULE algorithm. Moreover, we would like to delve into the concept of substituting ULE block in the second phase of GULE with a convex combination of optimal distributions from Section 2.5 (we mentioned it in Section 2.7.2).

Chapter 3

Big Data

3.1 Introduction to Reservoir sampling

Reservoir sampling is a randomized algorithm, which chooses some number of items from some population of unknown size N in a single pass over the items.[1] Remark that when the population is small, then all the items can be saved into the limited memory. Therefore, the arrangement of reservoir sampling makes sense only, when N is enormous and only some sample, of some given size k, of the population, can be saved. Let us highlight, that once the algorithm reveals an item, then it can be either saved or dismissed and the procedure cannot get another access to the dismissed element.

There are two types of reservoir sampling algorithm — *with* or *without replacement*. The first one allows to save the same item more than once, and the second restricts the sample to contain at most one instance of each item. Note that, when there is a reservoir sampling algorithm, which has a sample of size 1, then k independent repetitions of it give k-sample with replacement.

Without a loss of generality, we focus on the population in a form of an *online data stream* $\{S_t\}$. We assume that each item S_t is forwarded to the algorithm as an input in order compliant with this of indices t. The each data item from the stream can be given either in equitemporal steps or at any moment. Then t index can be associated with time of arrival of the data S_t. However, note that the data flows in some order, so t can be used to number the data in the order of revelations. In this dissertation we will assume the second of these approaches, so either $t \in [n]$ for some unknown number of items n, or simply $t \in \mathbb{N}$. We usually refer to them as *time steps*.

First, classical and simultaneously the most popular reservoir sampling algorithm is so called *Algorithm R*, attributed to Alan Waterman. The best known description of this procedure is provided by Vitter in the paper [105] from 1985.

[1] In fact, sampling is sometimes considered in more general arrangements. Especially when the number of items is know in advance, when the algorithm has instant access to any of the items from the population. However, these cases are not of our interest, so we constrain our definition. For more general solutions one can see e.g. [100].

A core idea of Algorithm R is as follows: we denote the stored element by *sample* and count the number of steps taken by n. After getting an element `item` from the data stream, we call the code from Algorithm 6.

ALGORITHM 6: Algorithm R

Data: a new data element `item`; present sampled element `sample`; counter n (initially 0)
Result: element `sample`

```
1 n = n + 1
2 if (random() < 1/n) then
3 |   sample = item
```

We use `random()`, which is pseudo-random generator of real numbers from the interval $[0, 1]$, mimicking a uniform random generator. Therefore, Algorithm 6 saves the n-th data (i.e. S_n) with probability $\frac{1}{n}$. One may easily check, that the index of saved element is uniformly distributed in $[n]$, where n is the present number of revealed items. Algorithm 6 provides 1-sample, however Vitter, in [105], has provided more general Algorithm R, which saves k-sample without replacements, where the number of indices of elements of the sample are uniformly distributed in $[n]$.[2]

Random sampling might be necessary in many situations where the amount of data generated (e.g. by the sensors) is huge. Streaming, collecting and analyzing all the data generated might be extremely ineffective due to the communication and computation overhead, while in many cases accurate diagnostics may be achieved basing on a relatively small data sample. One may especially wants to monitor the state of a cyber-physical system, where it is not necessary to analyze the whole historic data, but instead one may consider a sample drawn from a window of the last n values from the stream. For the same reason, it is not enough the keep only the last value(s), as then it might be impossible to observe dynamics of the system state.

Realize that, in order to capture the dynamics of the system, the statistics may require applying diverse probability distributions when choosing a random sample. For instance, we may need a higher precision of sampling for the most recent data items (high density of samples), and only a moderate accuracy for the older ones (a low number of samples). We should concern not only uniform samples, as in case of Algorithm R. Therefore, in this chapter, we focus on different distributions of random samples.

For a broad introduction to reservoir sampling algorithms, see [100].

3.2 Sliding Window model

[2] There are two changes: probability of a save is 1 until the k-th item of the stream and $\frac{k}{n}$ later, and also when one wants to save an obtained item, then it should uniformly at random substitute one of k already saved elements.

3.2.1 Introduction

In this part we concern a sampling data from an online data stream $\{S_t\}$ in a *sliding window* arrangement, introduced in 2002 by M. Datar, A. Gionis, P. Indyk, and R. Motwani in [31]. It concerns processing of a data stream where we are only interested in the most recent elements of the stream and the expired ones must not be taken into account. Such an approach is important for many applications (see, e.g., [6] and [32]), while some new applications may emerge, including, for instance, controlling cloud storage services (*proof-of-storage*). Necessity of using the sliding window model may also follow from legal regulations stating the data retention period in terms of a strict time limit.

We define sliding window as follows:

- $n \in \mathbb{N}$ is a fixed size of sliding window,
- at a moment t a sampler should hold a value S_{t+1-j}, where $j \in [n]$,
- regardless of t, $j \in [n]$ should be chosen according to a priori specified probability distribution D[3],
- the sampler has a constant memory while n might be large, so he forgets almost all of the recent n values of the stream except for a few ones.

The problem is that the window of the last n elements changes at each step and when we have to re-sample, then almost all values from which we have to choose are already forgotten.

A case of a sampler with constant memory and $D \sim \mathrm{Uni}\,(n)$ has been considered by Braverman, Ostrovsky and Zaniolo ([17]) in 2013. We present an alternative generic approach based on devil's staircase Markov chains. It generates a sample according to any admissible distribution D on the window of a size n and uses memory of size $\mathrm{O}\,(1)$. We provide a sufficient condition for the distribution to be admissible. While the class of admissible distributions is quite wide from the point of view of practical applications, we show some natural limitations for this class.

This section is devoted to one of the most fundamental processing primitives for sliding window processing — random sampling:

Definition 3.2.1 (Sliding window sampler)**.** *Given a window size n, a probability distribution D over $[n]$ and a data stream $\{S_t\}$, at time t, the sampler should return a single data item S_{t+1-j}, where $j \in [n]$ is chosen according to the probability distribution D. Especially, we focus on* constant memory *samplers that can store at most c values S_i at a time, where $c \in \mathbb{N}$ is a small constant.*

Construction of a sampler compliant with Definition 3.2.1 is a non-trivial task. Indeed, a sampled element must be eventually discarded – at the latest when it does not fit into the sliding window. At this moment the sampler has no

[3]Then j_t can be considered as a specific probabilistic counter (see Section 1.3.4), which takes values in $[n]$

choice and simply takes the most recent element from the stream as the sample value. The main problem is that the window of the last n elements changes at each step and when we have to re-sample, then almost all values from which we have to choose are already forgotten and we only can save the last observed element from the stream.

Note that Definition 3.2.1 concerns drawing a **single** stream item from the sliding window. In order to build a meaningful statistics it is almost always necessary to have more than one value. Therefore we are able to use several independent samplers described in Definition 3.2.1 to provide a bigger sample.

3.2.2 Previous constructions

The first non-trivial algorithm for sliding window sampling from a data stream appeared in the paper [7] from 2002 under the name *chain-sample* algorithm. A clear explanation of this algorithm and its properties can be found in [56]. The chain-sample procedure constructs a chain of moments, when a sample has to be modified. The expected length of such the structure is bounded by a constant e and with a high probability its length is bounded by $\mathrm{O}(\log n)$.[4] A non-constant size of the chain arise to be a serious disadvantage if the sampler must work extremely efficiently in order to process data arriving at a high rate.

In 2012, V. Braverman, R. Ostrovsky and C. Zaniolo in [17] published a constant memory sliding window sampler. Their algorithm is based on the following very simple, but clever idea. Namely, if U, V, W are finite and pairwise disjoint sets such that $|U| = |W|$, $X \sim \mathrm{Uni}(V \cup W)$, $Y \sim \mathrm{Uni}(U)$, where X and Y are independent, and Z is a random variable defined as

$$Z(\omega) = \begin{cases} X(\omega) & : X(\omega) \in V \\ Y(\omega) & : X(\omega) \notin V \end{cases}, \tag{3.1}$$

where $\omega \in \Omega$. The authors of [17], observed that then $Z \sim \mathrm{Uni}(U \cup V)$, what can be easily transformed into a simple sliding window sampler that divides the entire data stream into segments of length n (called buckets), constructs a uniformly distributed sample for each of the buckets using Algorithm 6 and returns one sample on the request amongst the last two samples using formula (3.1).

3.2.3 High level description of our contribution

We present a new approach to construct a constant memory sliding window sampler (according to Definition 3.2.1). First we show how to construct a random variable X with a distribution given a priori from so called strictly concave class of distributions on $\{1, \ldots, n\}$ in order to control the position of a sample in the stream. Further we provide admissible operations, which extend this class. For instance, our construction can be utilized in order to attain uniform sample, however it is different from the design of the algorithm from [17]:

[4] According to the authors, with probability at least $1 - n^{-c}$, for some constant c.

- we construct independent random variables X, Y from $[n]$ such that the random variable $\max(X, Y) \sim \text{Uni}\,(n)$,
- variables X and Y control the positions of the appropriate two samples from the stream
- $\max(X, Y)$ monitors the position of a uniform sample from the stream.

This way we maintain a uniform sample of size 1 over the sliding window of size n, satisfying constant memory requirements. The key point is that our algorithm, unlike the algorithm from [17], can be naturally extended to many non-uniform distributions. Another crucial advantage is that in our approach we never keep a sample that does not fall into the current sliding window (in [17] it is sometimes needed to outstay some elements for almost $2n$ steps,[5] while in [7] the relevant number of steps is even unbounded). This property can be important from a practical point of view, e.g. when a sliding window sampler with a strictly defined window size can be motivated by a legal requirement. In particular, the data in the stream may fall into the category of *personal data*. Consequently, authorization to process this data longer than for a stated step T may be forbidden.

Our algorithm is a derivative of Algorithm R (see Algorithm 6). A core idea of our algorithm is as follows: we denote the stored element by *sample* and use a probabilistic counter X ranged in the set $[n]$ in order to denote the position in the sliding window (i.e. at the moment t, sampler keeps the element S_{t+1-X}).[6] After getting an element `elem` from the data stream we call the update method Algorithm 7, which can change X and `sample`.

ALGORITHM 7: Basic reservoir sampling Algorithm

Data: sliding window's size n, a new data element `elem`
Result: pointer X; element `sample`

```
if (X == n) or (random() < q[X]) then
    X = 1
    sample = elem
else
    X = X + 1
```

Probabilities $q_i \in [0, 1]$, for $i \in [n-1]$,[7] are used to adjust the solution to the target probability distribution D of the pointer X. The method of initializing the variable `sample` and the counter X will be discussed later. Let us only remark that in any case the initial value of the counter X will be some number from the set $[n]$.

Notice that if $q_1 = q_2 = \ldots = q_{n-1} = 0$, then this method turns to a trivial cyclic refreshment method of the `sample` element from the stream – once the

[5] However, it can be easily rectified.
[6] See Section 1.3.4 for a definition of probabilistic counter.
[7] Note that the value of q_n does not matter.

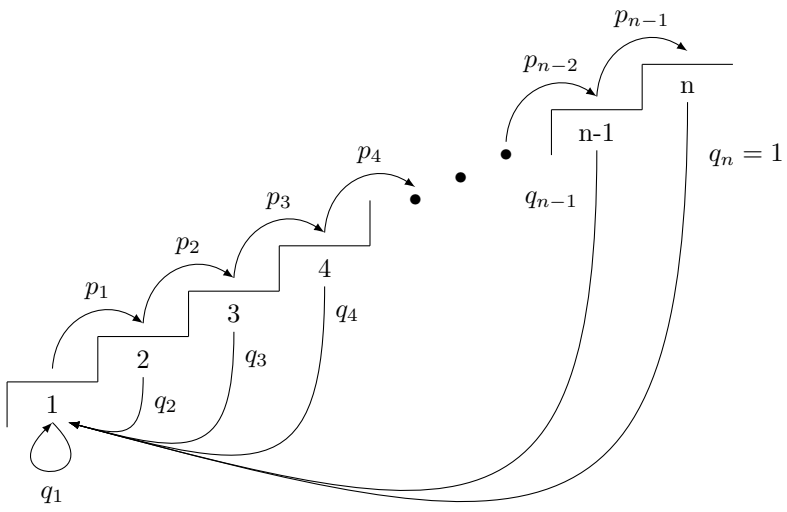

Figure 3.1: Schematic representation of Devil's Staircase. Arrows denote possible transitions between states together with theirs probabilities.

current sample leaves the sliding window, the freshly arrived stream element is taken instead. Similarly, if $q_i = 0$ for some i and the sampled element is the i-th recent one, then the new element in the stream will not replace the sampled one. On the other hand, if $q_i = 1$ for some $i \in [n-1]$, then `sample` will never store any element from positions $[i+1 : n]$ of the current sliding window and the window is effectively reduced to $[i]$. In order to exclude these degenerated cases, we assume that $0 < q_i < 1$ for all $i \in [n-1]$.

3.2.4 Devil's Staircase

Definition 3.2.2. *A Markov chain M on the set of states $[n]$ is called a devil's staircase*[8]*, if for $i \in [n-1]$ the only allowed transitions are from the state i either to the state $i+1$ or to the state 1. Moreover, for the state n the only transition is to the state 1. Let $p_i = \Pr[M(t+1) = i+1 | M(t) = i]$ and $q_i = \Pr[M(t+1) = 1 | M(t) = i]$. Then $p_i + q_i = 1$ for $i \in [n]$ and $q_n = 1$. We call the devil's staircase proper, if $0 < p_i < 1$ for each $i \in [n-1]$.*

In Figure 3.1, a schematic representation of such the Markov chain is presented. Arrows symbolize the transitions between the states.

Let us observe that the evolution of the random variable X from Algorithm 7 may be considered as a proper devil's staircase. Moreover, it is easy to see that

[8]Term according to [47], page 352.

each proper devil's staircase is ergodic, hence it has a stationary distribution (see e.g. [87] for basics of Markov chain theory).

Concavity of Cumulative Distribution Functions

A function $f : [0 : n] \to \mathbb{R}$ is *discrete concave* (see [66]) if

$$\frac{f(i+1) + f(i-1)}{2} \leqslant f(i) \tag{3.2}$$

for each $i \in [n-1]$. If this inequality is strict, then the function f is called *discrete strictly concave*. For the future use, note that the inequality $f(i+1) - f(i) \leqslant f(i) - f(i-1)$ is a convenient reformulation of the condition (3.2). Naturally, if f is discrete (strictly) concave on $[n]$, then its piece-wise linear extension to a function defined on the interval $[0, n]$ is also (strictly) concave.

Proposition 4. *Let $\pi = (\pi_i)_{i\in[n]}$ be a probability distribution on the set $[n]$. Let $F(k) = \sum_{i=1}^{k} \pi_i$ for $k \in [n]$ and $F(0) = 0$. Then the following conditions are equivalent:*

1. *π is a stationary distribution of some proper devil's staircase on the state space $[n]$;*
2. *F is strictly monotonic and discrete strictly concave.*

Proof. Let π be the stationary distribution of a proper devil's staircase Markov chain M with the transition probabilities defined as in Definition 3.2.2. We know that $0 < p_i < 1$ for all $i \in [n-1]$. Since π is stationary, we attain $\pi_{i+1} = \pi_i \cdot p_i$, hence $\pi_{i+1} < \pi_i$. As $F(i+1) - F(i) = \pi_{i+1} < \pi_i = F(i) - F(i-1)$, it follows that F is discrete strictly concave. Strict monotonicity of F is also clear, since $\pi_i > 0$ for all $i \in [n]$.

Suppose now that F is strictly monotonic and discrete strictly concave. Then $\pi_i > 0$ for $i \in [n]$. Strict concavity of F means that $\pi_{i+1} < \pi_i$ for $i \in [n-1]$. Let M be the proper devil's staircase Markov chain on $[n]$ with transition probabilities $p_i = \pi_{i+1}/\pi_i$ ($i \in [n-1]$) and let $\rho = (\rho_i)_{i\in[n]}$ be its stationary distribution. Then $\rho_{i+1} = \rho_i \cdot p_i = \rho_i \cdot \pi_{i+1}/\pi_i$. Consequently, $\rho_i = \rho_1 \cdot \pi_i/\pi_1$. Since $\sum_{i\in[n]} \rho_i = \sum_{i\in[n]} \pi_i = 1$, this implies that $\rho_i = \pi_i$ for $i \in [n]$. □

Let us note, that according to Proposition 4, we may provide another, short reformulation of the condition (3.2), in a language of stationary distribution, namely $\pi_i \geqslant \pi_{i+1}$.

Modelling Devil's Staircase

Let $F : [0 : n] \to [0, 1]$ be strictly monotonic, discrete strictly concave, $F(0) = 0$ and $F(n) = 1$. According to the proof of Proposition 4, F defines a devil's

staircase Markov chain M on the state space $[n]$, with the stationary distribution

$$\pi_k = F(k) - F(k-1) \tag{3.3}$$

and the probabilities p_k of transitions from the state k to $k+1$ given by the formula

$$p_k = \frac{F(k+1) - F(k)}{F(k) - F(k-1)} = \frac{\pi_{k+1}}{\pi_k} \tag{3.4}$$

for $k \leqslant n-1$. Note that discrete strict concavity of F is equivalent to a decreasing monotonicity of the sequence $(\pi_k)_{k\in[n]}$ (see remark after Eq. 3.2).

We can use Eq. (3.4) and Algorithm 7 to simulate the resulting Markov chain in order to build a sliding window sampler for window size n and CDF function F. This naive approach correctly sample the data according to F, however it requires generating a random real number at each step. From computational complexity point of view, such implementation would be inconvenient, especially for high rate data streams.

In order to reduce the computational complexity, we determine instead the time steps when transitions to the state 1 occur. This is enough, since we need to refresh the sampled values only at these moments. Periods between the time steps, in which the process returns to 1 are called *jumps*.

Let $s \in [n]$, and M be the resulting Markov chain which satisfies $M(0) = s$. Let us define a random variable

$$J_s = \min\{k \geqslant 1 : M(k) = 1\} \ .$$

Then J_s represents the number of steps that have to be taken between the current moment and the very next update of the `sample` variable, assuming that, currently the `sample` stores the s-th most recent element got from the data stream. Then, for $0 \leqslant k \leqslant n-s$, we have

$$\Pr[J_s > k] = \prod_{i=s}^{s+k-1} p_i = \prod_{i=s}^{s+k-1} \frac{\pi_{i+1}}{\pi_i} = \frac{\pi_{s+k}}{\pi_s} = \frac{F(s+k) - F(s+k-1)}{F(s) - F(s-1)}$$

and $\Pr[J_s > k] = 0$ for $k > n-s$. Therefore, for $k \leqslant n-s$,

$$\Pr[J_s \leqslant k] = 1 - \frac{F(s+k) - F(s+k-1)}{F(s) - F(s-1)} \tag{3.5}$$

and obviously $\Pr[J_s \leqslant n-s+1] = 1$.

Proposition 5. $\mathbb{E}[J_s] = \frac{1-F(s-1)}{\pi_s}$.

Proof.

$$\begin{aligned}\mathbb{E}[J_s] &= \sum_{k=0}^{n-s} \Pr[J_s > k] = \sum_{k=0}^{n-s} \frac{F(s+k) - F(s+k-1)}{F(s) - F(s-1)} \\ &= \frac{F(n) - F(s-1)}{F(s) - F(s-1)} = \frac{1 - F(s-1)}{\pi_s} \ .\end{aligned}$$

□

ALGORITHM 8: Basic update procedure

```
   procedure Init()
1  |  L = 0
2  |  sample = nil
3  |  sampleL = -1
4  |  state = random element from [n] chosen according to the stationary
   |    distribution of the Markov chain
5  |  z = random element from [n + 1 − state] chosen according to the
   |    distribution of the variable J_state
6  |  jump = state + z -1
   procedure Process(elem)
1  |  L++
2  |  state++
3  |  if state > jump then
4  |  |  state = 1
5  |  |  sampleL = L
6  |  |  sample = elem
7  |  |  jump = random element from [n] chosen according to the distribution
   |  |    of the variable J_1
   function Get()
1  |  return sample
```

Corollary 2. $E[J_1] = \frac{1}{F(1)}$.

3.2.5 A Fundamental Algorithm

Based on Eq. (3.5), we can build a quite effective sampling algorithm from the window of length n, according to the distribution with CDF F. Namely, we can simulate moves on the devil's staircase during reading data from the stream. We start from `state` $= 1$ and calculate a random element j_1 according to the distribution of J_1. Then at each step we increase `state` counter until it reaches j_1. At this moment we replace the sampled value by the current element from the stream, skip to the state 1, and again calculate j_1 according to the distribution of the variable J_1. We continue in this way as long as the elements from the string are coming in.

Another problem to solve is that the time of convergence of the considered Markov chain to the stationary distribution is quite long. A possible workaround for this issue is to start the process according to its stationary distribution. Notice that the algorithm constructed in this way will provide a sample after the first data will be stored according to the stationary distribution.

Our routine is described in Algorithm 8. It utilizes five variables:

- `sample` — which stores the actual sampled element from the data stream,
- `state` — which stores the current state of the devil's staircase,

- `jump` — which stores the next `state` after which the state of the devil's staircase would become 1,
- `sampleL` — which stores the position of the `sample` element in the data stream,
- `L` — which is a counter that stores the number of already observed elements from the data stream.

Let us remark that the last two variables are redundant, however they will be helpful for formulating an invariant used to show the correctness of our method.

First let us discuss the initialization of the sampler (the procedure `Init`). Its main objective is to start the process in the stationary distribution. For this purpose we may imagine that the stream has already started ($L = 0$) and this moment there exists a stored sample taken from the last n (imaginary, never seen) elements from the data stream. As a consequence, `sample` initially stores the value `nil` to imitate the stored imaginary element. The second issue is to initialize the position of this element in the sliding window of size n – we do it according to the stationary distribution and store the result in the variable `state`. Therefore we commence in a hypothetical situation in which the imaginary element is on the position in the sliding window indicated by the variable `state` and it has not been replaced yet by a newer sample. Note that the imaginary sample has to be replaced at the latest after $(n+1-\texttt{state})$ steps, while the number of steps to be executed before the update of `sample` is a random variable $J_{\texttt{state}}$. Therefore we choose a realization of the variable $J_{\texttt{state}}$ and add it to `state` in order to determine the moment when the transition to the state 1 should occur. The role of `jump` variable is to control `state`, in order to trigger the next replacement of the sampled element by the latest element from the data stream.

The main procedure `Process` of Algorithm 8 is executed when a new element from the data stream arrives. We simulate a move on the devil's staircase: First, we temporarily increment `state` by 1. We will leave it so unless the devil's staircase process has to return to the state 1. We test if it is the time to jump to the state 1 by checking the condition `state > jump` at each step. In case when this condition is satisfied, we substitute the variable `sample` with the current element of the stream, put `state = 1` and calculate the next `jump` position according to the distribution of the variable J_1. The function `Get` of Algorithm 8 is trivial: it returns the current sample element of the stream stored in the variable `sample`.

It is easy to check that since the first moment when we put `sample=elem` (line 6 of procedure `Process`), the equation `L-sampleL+1 = state` is satisfied, which shows that element `sample` is generated according to the stationary distribution of the devil's staircase. Moreover, let us observe that `L-1` $\in [n]$ when we assign `sample = elem` for the first time, so our algorithm surely returns a correct sample after reading the first n elements from the stream (remark that L is incremented before the aforementioned assignment).

Let us highlight that our solution is not based on a standard approach to irreducible Markov chains, which generally slowly converge to theirs stationary distributions (see e.g. [87] for a description of this approach). Indeed, our solution guarantees that the stationary distribution will be achieved after at most n steps.

3.2.6 Properties of the Devil's Staircase

By Proposition 4, the stationary distribution of a proper devil's staircase is described by a strictly monotonic and discrete strictly concave cumulative distribution function. Simple examples of such CDFs are functions $F(k) = \left(\frac{k}{n}\right)^{\alpha}$, for any $\alpha \in (0, 1)$. Indeed, the strict concavity of the function F results from the negativity of the second derivative of the natural continuous extension $\tilde{F}(x) = \left(\frac{x}{n}\right)^{\alpha}$ of F to a function on the interval $[0, n]$.

Note that when $\alpha = 1$, we get the CDF of the uniform distribution on $[n]$, which is not strictly concave, so Algorithm 8 and the devil's staircase cannot be used to get a sample with the uniform distribution in the sliding window. The next result shows a general limitation of strictly concave cumulative distribution functions.

Theorem 3.2.1. *Suppose that the CDF of a random variable X with values in $[n]$ is discrete strictly concave. Then*

$$\mathbb{E}[X] < \frac{n+1}{2} \ .$$

Proof. Let F be the CDF of X. and $\pi_i = F(i) - F(i-1)$. Since F is discrete strictly concave on $[n]$, we have $\pi_1 > \pi_2 > \ldots > \pi_n$. As $\sum_{i=1}^{n} \pi_i = 1$ we get immediately $F(k) > \frac{k}{n}$ for each $k \in [n-1]$. From this inequality and Eq. (1.1) we get that if X is a random variable with CDF F, then

$$\mathbb{E}[X] = \sum_{k=1}^{n} k(F(k) - F(k-1)) = n - \sum_{k=1}^{n-1} F(k) < n - \sum_{k=1}^{n-1} \frac{k}{n} = \frac{n+1}{2} \ .$$

□

3.2.7 Uniform Sample

The devil's staircase utilized in Algorithm 8 provides an elegant sampler, however it cannot yield a uniform sample and, by Theorem 3.2.1, no sample according to a probability distribution that is not biased towards the more recent values.

In this section we show that one can apply two devil's staircases to build a uniformly distributed sample from the sliding window of size n. In further sections we will show that this approach can be utilized more generally. The main idea behind our solution is based on the following observation:

ALGORITHM 9: Getting uniformly distributed sample

```
   procedure Init()
1  |  for i=1 to 2 do
2  |  |  sample[i] = nil
3  |  |  x = random()
4  |  |  X = ⌈nx^2⌉
5  |  |  state[i] = X
6  |  |  x = random()
7  |  |  C = (1 − x)(√X − √(X − 1))
8  |  |  jump[i] = min(n + 1, (1/4)(C + 1/C)^2) − 1
   procedure Process(elem)
1  |  for i=1 to 2 do
2  |  |  state[i]++
3  |  |  if state[i] > jump[i] then
4  |  |  |  sample[i] = elem
5  |  |  |  state[i] = 1
6  |  |  |  x = random()
7  |  |  |  jump[i] = min(n + 1, ⌈(1/4)(x(2 − x)/(1 − x))^2⌉) − 1
   function Get()
1  |  if state[1] > state[2] then
2  |  |  return sample[1]
3  |  else
4  |  |  return sample[2]
```

Proposition 6. *Suppose that X and Y are independent random variables with the same cumulative distribution function $F(k) = \sqrt{\frac{k}{n}}$, where $k \in [n]$. Then $Z = \max(X, Y)$ is uniformly distributed in $[n]$.*

Proof. It is enough to observe that, according to Section 1.3.2,

$$\Pr[Z \leqslant k] = \Pr[X \leqslant k] \cdot \Pr[Y \leqslant k] = \left(\sqrt{\tfrac{k}{n}}\right)^2 = \tfrac{k}{n} \ .$$

□

A distribution with CDF $F(k) = \sqrt{\frac{k}{n}}$ on $[n]$ we call *Square-Root distribution*, and denote by $\mathrm{SR}(n)$.

The Algorithm

Based on Proposition 6, our strategy is to run two independent instances of Algorithm 8 according to $\mathrm{SR}(n)$ distribution and derive the sample using the property from Proposition 6. The pseudo-code of our algorithm is presented in Algorithm 9.

The general main points of Algorithm 9 are as follows:

1. We run in parallel two independent instances of Algorithm 8 according to $\mathrm{SR}(n)$ distribution ($F(k) = \sqrt{k/n}$, for $k \in [n]$) and thereby create two independent copies of the devil's staircase. These instances are described by the variables
   $(\mathtt{L}_i, \mathtt{sampleL}_i, \mathtt{state}_i, \mathtt{jump}_i, \mathtt{sample}_i)$ for $i = 1, 2$. Since always $\mathtt{L}_1 = \mathtt{L}_2$, we replace them by a single variable $\mathtt{L}$.

2. As there are no interaction between the instances of devil's staircases, the variables $\mathtt{state}_1$ and $\mathtt{state}_2$ are independent and both follow the stationary distribution $\mathrm{SR}(n)$ of the modelled Markov chain.

3. Once $\mathtt{sample}_1 \neq \mathtt{nil}$ and $\mathtt{sample}_2 \neq \mathtt{nil}$, then $\mathtt{L} - \mathtt{sampleL}_1 + 1 = \mathtt{state}_1$ and $\mathtt{L} - \mathtt{sampleL}_2 + 1 = \mathtt{state}_2$, so both $\mathtt{state}_i$ follows the same stationary distribution $\mathrm{SR}(n)$ (independently).

4. By Proposition 6, the random variable $U = \max(\mathtt{state}_1, \mathtt{state}_2)$ is uniformly distributed in $[n]$. Therefore if $U = \mathtt{state}_1$, then our algorithm returns $\mathtt{sample}_1$, otherwise it returns $\mathtt{sample}_2$.

Similarly as in Algorithm 8, variables $\mathtt{L}$ and $\mathtt{sampleL}_i$, for $i = 1, 2$, are redundant in Algorithm 9.

Below we explain some details of the aforementioned approach.

During `Init` procedure we need to initialize $\mathtt{state}_i$ according to the stationary distribution of the devil's staircase. Note that for $x \in (0, 1)$ we have $\min\left\{k : x \leqslant \sqrt{k/n}\right\} = \lceil nx^2 \rceil$. This observation is used in lines 2-4.

The next task is to generate a realization of random variable J_s, where s is given by $\mathtt{state}_i$. Here the situation is slightly more complicated. For a given $x \in (0, 1)$ we have to find minimal k such that $x \leqslant \Pr[J_s \leqslant k]$. For $k = n - s + 1$, this inequality always holds. By Eq. (3.5), if $k \leqslant n - s$, then $x \leqslant \Pr[J_s \leqslant k]$ translates to $\frac{(\sqrt{s+k} - \sqrt{s+k-1})}{(\sqrt{s} - \sqrt{s-1})} \leqslant 1 - x$, that is

$$\sqrt{s+k} - \sqrt{s+k-1} \leqslant (1-x)(\sqrt{s} - \sqrt{s-1}) \ . \tag{3.6}$$

Let $C = (1-x)(\sqrt{s} - \sqrt{s-1})$. We can simplify (3.6) to $\sqrt{s+k} - \sqrt{s+k-1} \leqslant C$. After squaring both sides, reduction of the common terms and some simplifications, we get $s + k - \sqrt{(s+k)(s+k-1)} \leqslant \frac{1}{2}(C^2 + 1)$ and $s + k - \frac{1}{2}(C^2 + 1) \leqslant \sqrt{(s+k)(s+k-1)}$. After squaring both sides again, a few simplifications and taking into account that $\Pr[J_s \leqslant n - s + 1] = 1$, we obtain the final formula:

$$\min\{k : x \leqslant \Pr[J_s \leqslant k]\} = \min\left(n+1, \left\lceil \tfrac{1}{4}\left(C + \tfrac{1}{C}\right)^2 \right\rceil\right) - s \ , \tag{3.7}$$

used in lines 6-8 of `Init`. Notice that, according to the definition of J_s, the first transition to the state 1, starting from the state s will take place after the variable `state` reaches the value $s + J_s - 1$. For $s = 1$, Eq. (3.7) simplifies to the following form used in the lines 6-7 of `Process`:

$$\min\{k : x \leqslant \Pr[J_1 \leqslant k]\} = \min\left(n+1, \left\lceil \tfrac{1}{4}\left(\tfrac{x(2-x)}{1-x}\right)^2 \right\rceil\right) - 1 \ .$$

Use of Randomness

There are two places in Algorithm 9 where we call pseudo-random generator. First time we use it twice in the initialization phase. In the procedure `Process` the random number generator is called once when the variable `state` exceeds the value `jump`. From Corollary 2, we get $\mathbb{E}[J_1] = (F(1))^{-1} = \sqrt{n}$. Hence the expected time period between successive calls to the pseudo-random number generator by each copy of modelled devil's staircase is $\sqrt{n}$.

Remark that the algorithm provided by Braverman et al. in [17] can be analogically adapted to a version with jumps in order to reduce a number of updates. Such an adaptation has smaller number of expected updates than Algorithm 9 for uniform distribution in the sliding window. Indeed, the i-th element from a bucket is saved with probability $\frac{1}{i}$, so there are H_n jumps within every bucket of size n on average. However, we have to bear in mind that the adaptation of algorithm from [17] cannot be naturally generalized to any other distributions, conversely to our approach. Since it is applicable only for a specific distribution, it is quite common, that it performs better than the general solution in the same arrangement. As we mentioned in Section 3.1, non-uniform solutions may be significant from a point of applications.

3.2.8 Examples of Non-uniform Sampling

Examples of discrete strictly concave distributions on $[n]$

We now turn our attention to the class of probabilistic distributions that can be modelled using the single devil's staircase. From Section 3.2.6 we know that if the CDF F is discrete strictly concave on $[n]$, then F is a stationary distribution of some devil's staircase on $[n]$. We present several known examples of such functions:

- functions $F_\alpha(k) = \left(\frac{k}{n}\right)^\alpha$, for $\alpha \in (0,1)$ and $k \in [n]$,
- R-shifted version of geometric distributions, conditionally truncated to $[n]$, i.e. $(\mathrm{Geo}_\rightarrow(p)|[n])$ — the probability mass functions of these random variables are given by the formula $\Pr[X = k] = \frac{pq^{k-1}}{1-q^n}$, where $k \in [n]$ (see Section 1.3.10),
- Zipf distributions $Z(n,m)$— the probability mass functions of these random variables are given by the formula $\Pr[X = k] = \frac{1}{k^m H_n{}^{(m)}}$, for $k \in [n]$ (see Section 1.3.11).

Any distribution listed above can be realized with one copy of the devil's staircase and all formulas required for the implementation can be derived with a little effort (what will be presented further). As we have seen, there are many probability distributions that do not fall to this category, with the prominent example of the uniform distribution.

Conditionally truncated examples

Consider a class $\mathcal{N}$ of distributions with support $\mathbb{N}$, whose CDFs are strictly monotonic and discrete strictly concave. For a given $n \in \mathbb{N}$, we may define a class $\mathcal{N}_n$ of distributions from $\mathcal{N}$ conditionally truncated to $[n]$, namely

$$\mathcal{N}_n := \left\{ F_{(n)} : [n] \to [0,1] : F \in \mathcal{N}, F_{(n)}(k) = \frac{F(k)}{F(n)} \right\} .$$

Note that every function $F_{(n)}$ from $\mathcal{N}_n$ is strictly rising and discrete strictly concave, because $F(k+1) > F(k)$ and $F(k+1) - F(k) > F(k) - F(k-1)$ for any $k \in [n-1]$.

Let us realize that both $Z(n,m)$ (for $m > 1$) and $(\mathrm{Geo}_{\to}(p)|[n])$ belong to $\mathcal{N}_n$.

Just to mention a few other examples from this class, let us enumerate: a diadic distribution – with pdf given by $\Pr[X = k] = \frac{2^{-k}}{1-2^{-n}}$, a logarithmic distribution – with pdf given by $\Pr[X = k] = \frac{1}{a_n} \frac{p^k}{-k \ln(1-p)}$, where a_n is a normalizing constant given by the first n non-zero terms of Maclaurin series of $-\ln(1-p)$ with respect to p, or $(\mathrm{Pois}_{\to}(\lambda)|[n])$ (for $\lambda \leqslant 1$) – with pdf given as $\Pr[X = k] = \frac{1}{a_n} \frac{\lambda^{k-1}}{(k-1)!} e^{-\lambda}$, where a_n is a normalizing constant, given by the first n non-zero terms of Maclaurin series of e^{λ} with respect to λ.

Derivations of expected jumps

According to Eq. (3.5), when we want to generate a realization of random variable J_s for a given $X \sim F_{(n)} \in \mathcal{N}_n$, we need to find:

$$\min \left\{ k \in [n-s+1] : 1 - x > \Pr[J_s > k] = \frac{F_{(n)}(k+s) - F_{(n)}(k+s-1)}{F_{(n)}(s) - F_{(n)}(s-1)} \right\} .$$

Realize that we can transform the internal inequality to

$$\Pr[X = s] = (F(s) - F(s-1)) > \frac{F(k+s) - F(k+s-1)}{(1-x)} = \frac{\Pr[X = k+s]}{(1-x)} , \tag{3.8}$$

for $k \in [n-s]$ and $0 < x < 1$, and note that $1 - x > \Pr[J_s > n - s + 1] = 0$ is trivially fulfilled for $x \in (0,1)$.

Let us consider $X \sim Z(n,m)$ at first.

Proposition 7. *$Z(n,m)$ is discrete strictly concave on $[n]$.*

Proof. Let $X \sim Z(n,m)$ and $i \in [n-1]$. Then

$$\Pr[X = i] = \frac{1}{i^m H_n^{(m)}} > \frac{1}{(i+1)^m H_n^{(m)}} = \Pr[X = i+1] .$$

□

Now we consider how we can simply generate a jump J_s related to $Z(n,m)$ distribution. According to Eq. (3.8),[9] we are searching for

$$\min\left\{k \in [n-s+1] : (k+s)^m > \frac{s^m}{(1-x)}\right\} \ .$$

The internal inequality can be easily transformed to $k \geqslant \frac{s}{\sqrt[m]{1-x}} - s$. Therefore, in order to generate a realization of J_s for devil's staircase with $Z(n,m)$ distribution, we need to generate $x \in \mathrm{Uni}\,(0,1)$ and then put

$$j_s = \min\left(n+1, \left\lceil \frac{s}{\sqrt[m]{1-x}} \right\rceil\right) - s \ .$$

Thanks to Proposition 5, we know that $\mathbb{E}[J_s] = s^m\left(H_n^{(m)} - H_{s-1}^{(m)}\right)$. Especially, when $s=1$, we have $\mathbb{E}[J_1] = H_n^{(m)}$. However, notice, that for $m>1$, $\mathbb{E}[J_1] < \zeta(m) = \mathrm{O}\,(1)$.[10] Especially, when $m \geqslant 2$, then $\zeta(m) \leqslant \frac{\pi^2}{6} = 1.6449\ldots$, hence the reduction by jumps is so meager, it does not make much sense in this case. However, when $m=1$, then $\mathbb{E}[J_s] = s(H_n - H_{s-1}) \approx s\ln\left(\frac{n}{s-1}\right)$ for $s>1$ and in particular $\mathbb{E}[J_1] = H_n \approx \ln(n) + \gamma$. In this special case, the skipping is more practical.

Now let us consider $X \sim (\mathrm{Geo}_{\rightarrow}(p)|[n])$.

Proposition 8. *$(\mathrm{Geo}_{\rightarrow}(p)|[n])$ is discrete strictly concave on $[n]$.*

Proof. Let $X \sim \mathrm{Geo}\,(p,n)$ and $i \in [n-1]$. Then

$$\Pr[X=i] = \frac{q^{i-1}p}{1-q^n} > \frac{q^i p}{1-q^n} = \Pr[X=i+1] \ .$$

□

We may relatively easy generate a jump J_s related to $(\mathrm{Geo}_{\rightarrow}(p)|[n])$ distribution. Indeed, according to Eq. (3.8), we are searching for

$$\min\left\{k \in [n-s+1] : \frac{(1-p)^{k+s-1} - (1-p)^{k+s}}{(1-p)^{s-1} - (1-p)^s} < 1-x\right\} \ .$$

Now we can briefly see that the left hand side of the inequality in the condition of the above minimum is simply

$$\frac{(1-p)^{s-1}\left((1-p)^k(1-(1-p))\right)}{(1-p)^{s-1}(1-(1-p))} = (1-p)^k \ ,$$

so we obtain $k > \frac{\ln(1-x)}{\ln(1-p)}$. Therefore, in order to generate J_s for this distribution, we need to generate $x \in \mathrm{Uni}\,(0,1)$ and then put

$$j_s = \min\left(n+1-s, \left\lceil \frac{-\ln(1-x)}{-\ln(1-p)} \right\rceil\right) \ .$$

[9] Note that this formula also applies for $m=1$.

[10] See Section 1.2.10 for definition of Riemann's Zeta function.

Let us note, that this approach is not as problematic for very small parameter p as the one considered in Section 2.8.4.

Thanks to Proposition 5, we know that

$$\mathbb{E}[J_s] = \left(1 - \frac{1-q^{s-1}}{1-q^n}\right)\frac{1-q^n}{pq^{s-1}} = \frac{1-q^{n-s+1}}{p} \ ,$$

what for $s = 1$, reduces to $\mathbb{E}[J_1] = \frac{1-q^n}{p}$. Remark, that when $p = \frac{c}{n}$ for some constant $c \ll n$, then $\mathbb{E}[J_1] \approx \frac{n(1-e^c)}{c}$. In this case, queries send to pseudo-random generator are very rare.

Now, let us concern a diadic distribution. In this case, Eq. (3.8) provides simply that we are searching for

$$j_s = \min\left\{k \in [n-s+1] : 2^{-k} < 1-x\right\} = \min\left(n-s+1, \lceil -\ln(1-x)\rceil\right).$$

Moreover, from Proposition 5 we get

$$\mathbb{E}[J_s] = \left(1 - \frac{1-2^{-s+1}}{1-2^{-n}}\right)\frac{1-2^{-n}}{2^{-s}} = \frac{2^{-s+1}-2^{-n}}{2^{-s}} = 2 - 2^{-n+s} \ ,$$

which approximately equals 2, hence we obtained similar result to this for Zipf distribution for $m \geqslant 2$.

Let us consider a logarithmic distribution. By Eq. (3.8), we want to find

$$\min\left\{k \in [n-s+1] : \frac{p^{k+s}}{k+s} < (1-x)\frac{p^s}{s}\right\}.$$

Using positive branch of W-Lambert function (see Section 1.2.12) in order to obtain:

$$j_s = \min\left(n+1, \frac{W_0\left(\frac{-\ln(p^s)}{p^s(1-x)}\right)}{-\ln(p)}\right) - s \ .$$

With Proposition 5 it is possible to obtain that

$$\mathbb{E}[J_s] = \sum_{i=0}^{n-s}\frac{sp^i}{s+i} \ .$$

Let us mention, that it is to provide simple analogous formulas for R-shifted Poisson distribution, conditionally truncated to $[n]$, hence we omit them.

3.2.9 Extension with m-root strictly concave functions

First we generalize the method used for solving the case of the uniform distribution.

Definition 3.2.3. *A CDF $F : [n] \to [0,1]$ is m-root strictly concave, if the function $\sqrt[m]{F}$ is discrete strictly concave.*

If $m \in \mathbb{N}$ and CDF F is m-root discrete strictly concave, then we can build a sampler which utilizes m independent instances of the devil's staircase with the stationary distribution $\sqrt[m]{F}$ in order to provide the distribution F on the sliding window of size n. Correctness of this solution is based on the following fact:

$$\Pr[\max(X_1, \ldots, X_m) \leqslant x] = \left(\sqrt[m]{F(x)} \right)^m = F(x) \ .^{11}$$

Natural examples of such CDFs are of the form $F_\alpha(k) = \left(\frac{k}{n}\right)^\alpha$ for $\alpha \in (0, m)$. Samplers with cumulative distribution functions F_α for $\alpha > 1$ may be used in the situations when the old values from the stream are more interesting than the current ones.

The following result gives natural bounds on the class of probability distributions, which can be modelled by the aforementioned method:

Theorem 3.2.2. *If the CDF of a random variable X is m-root discrete strictly concave on $[n]$, then*

$$\mathbb{E}[X] < \frac{m}{m+1} n + \frac{1}{2} + \mathrm{O}\left(\frac{1}{n}\right) \ .$$

Proof. A proof of this result is analogous to the one of Theorem 3.2.1.

Let F be the CDF of X and let $H = \sqrt[m]{F}$. Then H is discrete strictly concave on $[n]$, so H has a piece-wise linear extension to a strictly concave function $\tilde{H}$ on the interval $[0, n]$. Therefore, for each $x \in [0, n]$ we have

$$\tilde{H}(x) = \tilde{H}\left(\left(1 - \frac{x}{n}\right) 0 + \frac{x}{n} n\right) > \left(1 - \frac{x}{n}\right) H(0) + \frac{x}{n} H(n) = \frac{x}{n} \ .$$

Hence $F(x) > \left(\frac{x}{n}\right)^m$. From this inequality, Eq. (1.1) and the classical Faulhaber's Formula (1.13), we get

$$\begin{aligned} \mathbb{E}[X] &= \sum_{k=1}^{n} k(F(k) - F(k-1)) = n - \sum_{k=1}^{n-1} F(k) \\ &< n - \sum_{k=1}^{n-1} \left(\frac{k}{n}\right)^m = \frac{m}{m+1} n + \frac{1}{2} + \mathrm{O}\left(\frac{1}{n}\right) \ . \end{aligned}$$

□

The following simple observation provides the monotonicity of m-root strict concavity property:

Proposition 9. *If f is m-root (strictly) concave, then it is $m+1$-root (strictly) concave.*

Proof. Observe that $f^{\frac{1}{m+1}} = (f^{\frac{1}{m}})^{\frac{m}{m+1}}$ and the function $g(x) = x^{\frac{m}{m+1}}$ is concave on $(0, \infty)$. Moreover, a superposition of two concave functions is concave, what ends the proof. □

[11] According to Section 1.3.2.

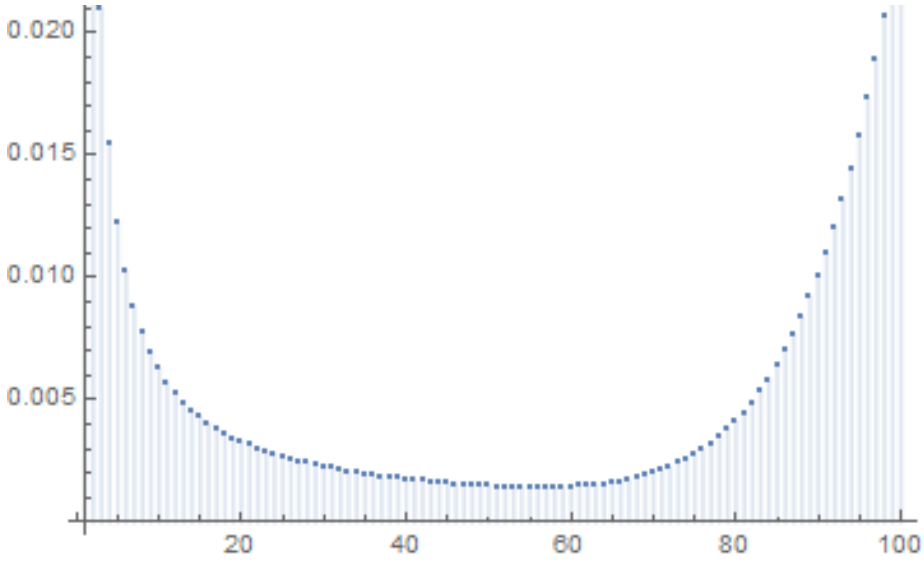

Figure 3.2: An example of U-shaped probability mass function associated with cumulative distribution function: $0.75F_{0.1} + 0.25F_{10}$ for $n = 100$.

Proposition 9 bears the following observation: if F_X is m-root strictly concave for some $m \in \mathbb{N}$, but it is not $m-1$-root strictly concave, then one should use at most m independent devil's Markov chains in order to generate X. Nevertheless, for many CDFs (for $m \geqslant 3$), with such the property, we have not been able to determine whether it is possible to provide appropriate samplers, which utilize less than m independent random variables or not. It is quite challenging, open problem.

3.2.10 Other extensions

Convex combinations

Let us finally observe that from independent samplers with CDF functions F and G one can construct samplers for any convex combination $\alpha F + \beta G$ where $\alpha, \beta > 0$ and $\alpha + \beta = 1$. For example, the probability mass function related to cumulative distribution function $0.75F_{0.1} + 0.25F_{10}$ is U-shaped as presented in Figure 3.2.[12] In this case we run two samplers – one for $F_{0.1}$ and one for F_{10} – and finally output the sample provided by the first sampler with probability 0.75 and by the second sampler with probability 0.25. Note that this observation can be used for any finite number of independent samplers, which a reservoir sampling algorithm can fit into memory.

[12]This kind of distributions are called bathtub curve distributions and are related e.g. to the failure rates over time in the reliability theory ([75]).

Order statistics solution

Assume for a while, that we have m independent samplers, modelled by variables $X_1, \ldots, X_m$. Then, one can form a (potentially) new sampler, modelled by any order statistic $\mathrm{Ord}_{i:m}(X_1, \ldots, X_m)$, for $i \in [m]$. Note that, when $i = m$, then this method provides $\max(X_1, \ldots, X_m)$, so it can be identified as a generalization of the m-root strictly concave approach. Especially, one may utilize $\min(X_1, \ldots, X_m)$ (a case $i = 1$). A correctness of this approach can be justified by the fact, that by operation $\mathrm{Ord}_{i:m}$ we can only obtain positions from the set $\{X_1, \ldots, X_m\}$, so such the new sampler has an access to the sample from the appropriate position, which corresponds to the variable, which realized the order statistic.

3.2.11 A class of $\mathcal{DS}$-admissible distributions

We call a distribution $\mathcal{DS}$-admissible, if it can be modelled by a finite number of devil's staircases.[13]

Naturally, discrete strictly concave distributions are $\mathcal{DS}$-admissible. They are applicable, for instance, when one wishes to sample mostly the recent elements from the stream.

We have seen in Section 3.2.9 that m-root strictly concave distributions are $\mathcal{DS}$-admissible and are produced via max operation from m copies of i.i.d. variables with discrete strictly concave distribution. The subclasses of m-root strictly concave distributions (for $m \leqslant 2$ are designated to sample more often the older elements from the sliding window. If such the phenomenon is preferred, then the generalization described in Section 3.2.9 should be applied.

As provided in Section 3.2.10, one can also utilize other order statistics like e.g. min.

In some applications it may be justified to use e.g. U-shaped distributions, which can be obtained by convex combinations of m-root strictly concave distributions (see Section 3.2.10).

Therefore $\mathcal{DS}$-admissible class is closed under the operations of order statistics $\mathrm{Ord}_{i:m}$ (in particular max and min operations) and convex combinations.[14] The class of discrete strictly concave distributions plays a role of a generator of $\mathcal{DS}$-admissible class, i.e. every distribution from $\mathcal{DS}$-admissible class can be achieved from some finite number of discrete strictly concave distributions by the allowed operations.

However, finding an elegant characterization of the class of $\mathcal{DS}$-admissible distributions is an open problem. Also a very interesting and challenging problem can be a way of an extension of this class by allowing operations, which do not assume the independence of the variables.

[13] Note that $\mathcal{DS}$ is an abbreviation of devil's staircase.

[14] We have been assuming the independence of the variables, on which the aforementioned operations act.

3.2.12 Application

We are going to analyze a dataset of size 2500 of shares' prizes of General Electric in a period from 30-12-2010 to 5-12-2020 (one data item per day). Let us consider a sliding window of size $n = 1500$ in two arrangements: the first one – according to $\mathrm{SR}(n)$ distribution, and the second – according to $\mathrm{Uni}\,(n)$ distribution in the sliding window of size n. In both cases we utilize $k = 250$ independent copies of appropriate reservoir sampler, in order to obtain a sample with replacements.

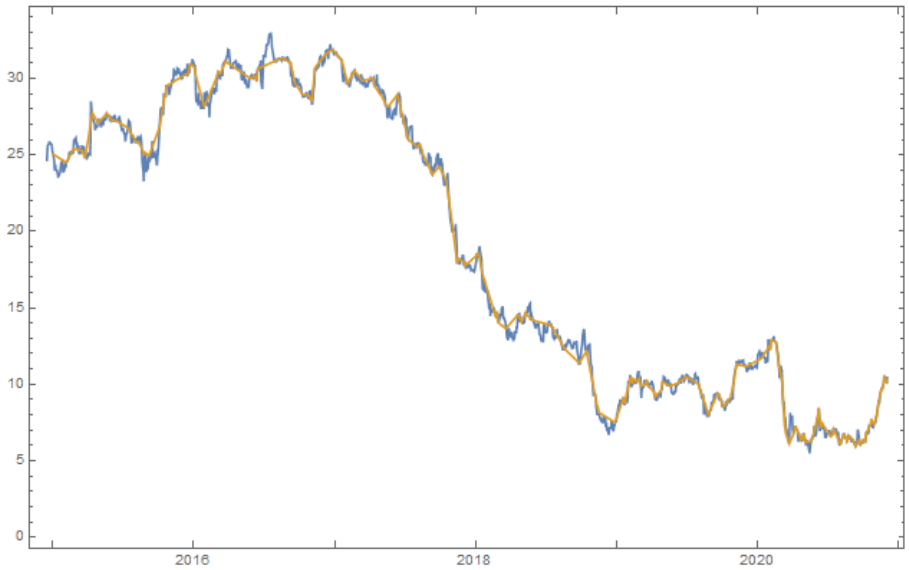

Figure 3.3: Precision of an approximation of General Electric shares' prizes by 250 independent copies of reservoir sampler, according to SR(1500) distribution.

In Figure 3.3 we present all the shares from a period from 17-12-2014 to 5-12-2020 (1500 data items, depicted in blue) juxtaposed with 250 single samples, given by our algorithm (discrete strictly concave case), which samples from SR(1500) distribution in that period. In order to provide approximation, we use linear interpolation of the obtained samples, together with two additional points, given by samples saved on 17-12-2014 and 5-12-2020. This approximation is presented in orange.

In Figure 3.4 we can see an analogous plot of the same shares' prizes juxtaposed with 250 single samples, but provided by our algorithm (2-root strictly concave case), which samples from $\mathrm{Uni}\,(1500)$ distribution in the aforementioned period. Again, a linear interpolation of the samples is depicted in orange.

It is visible that the approximation in Figure 3.3 is very precise for the most recent period and there are merely several data items sampled from 2015 year, which provides the garbled interpolation of the data from this period. This ap-

proach is definitely desirable when one wants to analyze the data from the given sliding window, basing mainly on the recent items and the plenty of old realizations. Similarly, it is clear from Figure 3.4 that the second approach provides the relatively good approximation over the whole sliding window. However, as we can see in Figure 3.4, in the first semester of 2020 (hence in the recent period), the interpolation omitted some quite significant fluctuations of the shares' prizes. This affirms the statement, that in cases, when the recent data items have higher priority than the others, then the approach with uniformly distributed sample is insufficient.

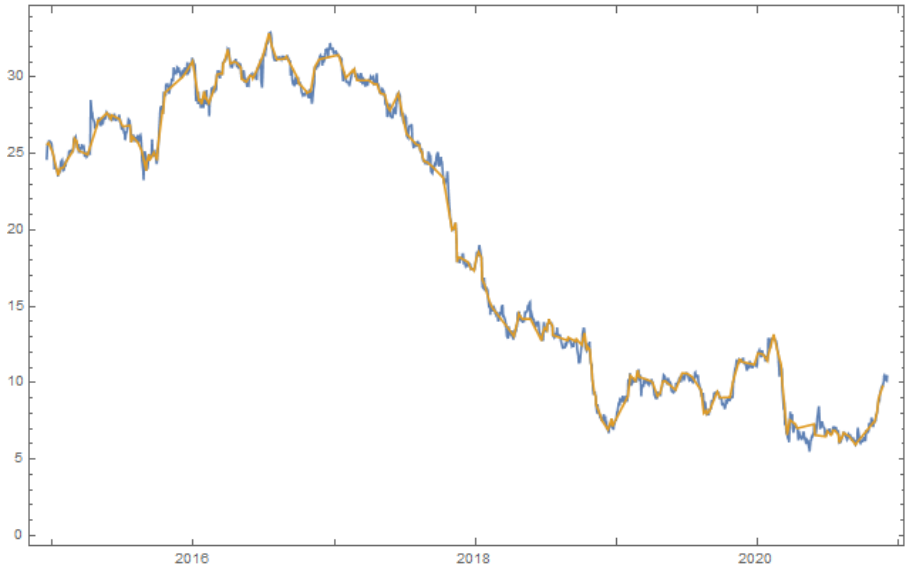

Figure 3.4: Precision of an approximation of General Electric shares' prizes by 250 independent copies of reservoir sampler, according to Uni (1500) distribution.

3.2.13 Final remarks

We have presented a collection of methods that together with the paradigm of the devil's staircase can be used to design sliding window samplers with different probability distributions for the position of the sample.

Our solution is more flexible than the one proposed in [6]. However, in the special case of uniform distribution it uses more calls to pseudo-random number generator than the solution from [6].

The effectiveness of our implementation will depend on the computational complexity of the procedure of determining the smallest number k such that $\Pr[J_s \leqslant k] > x$ (see Eq. (3.5)) for $x \in (0, 1)$.

As mentioned in Section 3.1, any of described methods, which provide the

single elements with positions in the sliding window provided randomly according to some $\mathcal{DS}$-admissible distributions, can be simply extended to produce r-sample in the sliding window by running r independent copies of the appropriate algorithm. Of course, this method produces a sample of r elements with replacements.

A memory cost of our method is linear with respect to the (constant) size of the sample and is only dependent on the distribution of positions of the sample in the sliding window. Especially, Algorithm 9 performs similarly to the state-of-the-art algorithm from [17] (in the sense of the size of used memory).

3.3 Power Law of Update

The foregoing part is devoted to a certain class of probabilistic snapshots for elements of the observed data stream. We show how one can control their probabilistic properties with probabilistic counters (see Section 1.3.4) and present some of the potential applications. Our solution can be used to store information from the observed history with limited memory (similarly as in sliding window model, but without a constraint of the expiration of some elements after some time. It can be used for both web server applications and Ad Hoc networks. For example, for automatic snapshots taking from online video stream of unknown size.

3.3.1 Introduction

Suppose that we are observing an online and infinite data stream $\{S_t\}_{t=1}^{\infty}$. Our goal is to keep an element from this stream with a prescribed position.[15] For instance, we may be interested to keep the element S_i, with index i being close to $\lfloor \frac{n}{2} \rfloor$ after reading first n elements from the stream. Of course this problem is trivial if we have a direct access to all elements $S_1, \ldots, S_n$ at any time (in particular, at the moment when S_n is read). However, in our arrangement, we deal with large amount of data and keeping all the information into memory is exaggeratedly expensive and thence undesirable. We have to keep in mind that our memory resources are substantially limited.

In this section we investigate plenty of randomized procedures which intuitively allow us to choose elements located near the required position in the stream of data. All these procedures are based on the same schema. Fundamentally, they only differ on a choice of a sequence (α_n) of probabilities which are used to control updates of reservoir sample. In this section we consider a case of sample of size 1. However, according to the previous arguments (e.g. from Section 3.1), one can easily provide a k-sample with replacements via independent copies of the algorithm. More precisely, α_n is the probability, that S_n will be saved in the reservoir at the time step, when it arrives. Naturally, we do not want to keep blank samples, so the first revealed data provided by the stream is always saved, hence $\alpha_1 = 1$.

[15] Similarly to the approach of Section 3.2, but without the sliding window constraint.

ALGORITHM 10: Reservoir sampling algorithm according to update sequence $(\alpha_n)_n$

```
   procedure Init()
1  |  K= 0
2  |  n= 0
3  |  sample= nil
   procedure Process(elem)
1  |  n= n+1
2  |  if random() ⩽ α[n] then
3  |  |  K= 1
4  |  |  sample = elem
5  |  else
6  |  |  K= K+1
   function Get()
1  |  return sample
```

In a description of our Algorithm 10 we utilize three variables:

- K – which stores present value of probabilistic counter,
- n – which is the number of elements of the stream, which have been already read,
- `sample` – which stores presently sampled element in the reservoir of size 1 — a probabilistic snapshot.

and an *update sequence* α_n of transition probabilities.

Algorithm 10 is a modification of classical Algorithm 6:

Initialization sets the appropriate values of three utilized variables before revealing the first data item. In the procedure Process we use `random`() function which is a high quality pseudo-random number generator of real numbers from the interval $[0, 1]$. The function $\alpha[n]$ represents an update sequence $(\alpha_n)_n$ of probabilities of saving the n-th data item provided by the stream. Note that K is either reset to 1 every time a data item is saved into the reservoir. Otherwise it is increased by 1. It can be interpreted as a discrete process, similar to devil's staircase (see Definition 3.2.2), but with $\mathbb{N}$ as a support and with transition probabilities dependent on the present time step. Therefore it can be interpreted in terms of probabilistic counter (see Section 1.3.4). Note that K is utilized to control the position of the present sample in the stream.

Connection with Algorithm R

Suppose that we use Algorithm 10 with the update sequence $\alpha_n = \frac{1}{n}$. This way, we obtain the classical Vitter's Algorithm R (Algorithm 6) published in [105]. Let K_n denotes the value of the random variable K after reading n data items from the stream. It is well known that in this case (i.e. when $\alpha_n = \frac{1}{n}$) we

have $\Pr[K_n = i] = \frac{1}{n}$ for each $i \in [n]$. In other words, $K_n \sim \mathrm{Uni}\,(n)$. Therefore $\mathbb{E}[K_n] = \frac{n+1}{2}$ (see Section 1.3.6).

3.3.2 Power Law of Update

In this part we present, so called Power Law of Update model (or shortly PLU model). PLU implies the special form of update sequences $(\alpha_n)_n$. Namely, $\alpha_n = \min\left(1, \frac{g}{n^\alpha}\right)$, where $\alpha \geqslant 0$ and $g > 0$ are some fixed parameters.[16] PLU owns its name to the expression of n^α in the denominator of the crucial part of the formula for α_n. This approach allows to apply appropriate update sequence in order to control the history of a massive streams of data. The aforementioned case $\alpha_n = \frac{1}{n}$ (i.e. update sequence for Algorithm R) is also a very special case of PLU approach to Algorithm 10.

In a series of subsections we will analyze a behaviour of the random variable K_n for different update sequences in PLU model. Beneath we present a summary of our results:[17]

- if $\alpha = 0$ and $g \in (0,1)$, then the random variable $K_n \xrightarrow[n\to\infty]{d} \mathrm{Geo}_{\to} g$, hence $\mathbb{E}[K_n] \sim \frac{1}{g}$,
- if $\alpha \in (0,1)$, then the random variable $\frac{K_n}{n^\alpha} \xrightarrow[n\to\infty]{d} \mathrm{Exp}\,(g)$, hence $\mathbb{E}[K_n] \sim \frac{n^\alpha}{g}$,
- if $\alpha = 1$, then the random variable $\frac{K_n}{n} \xrightarrow[n\to\infty]{d} \mathrm{B}\,(1,g)$, hence $\mathbb{E}[K_n] \sim \frac{n}{g+1}$,
- if $\alpha \in (1,2)$, then $\mathbb{E}[L_n] = \frac{g}{2-\alpha} n^{2-\alpha} + \mathrm{O}\left(\max\left(1, 1_{\left\{\frac{3}{2}\right\}}(\alpha) \ln n, n^{3-2\alpha}\right)\right)$,
- if $\alpha = 2$, then $\mathbb{E}[L_n] = g \ln(n) + \mathrm{O}\,(1)$,
- if $\alpha > 2$, then $\mathbb{E}[L_n] = \mathrm{O}\,(1)$.[18]

3.3.3 General Properties

Let $(\alpha_n)_n$ be an update sequence ranged in the interval $[0,1]$ with starting condition $\alpha_1 = 1$. Moreover, let K_n be a sequence of consecutive integers tracing the value of the variable K from Algorithm 10 after n runs of Process with the update sequence α_n. Clearly, $K_n \in [n]$ for each n.

Notice that the interpretation of $(\alpha_n)_n$ instantly gives $\Pr[K_1 = 1] = 1$, $\Pr[K_n = 1] = \alpha_n$ and

$$\Pr[K_{n+1} = k+1] = (1-\alpha_n)\Pr[K_n = k] \ , \tag{3.9}$$

[16] Minimum operation is only to restrict the values to $[0,1]$, since α_n is a probability.

[17] For definitions of the convergence in distribution and the utilized distributions see Section 1.3.

[18] All $\mathrm{O}\,()$ and $\sim$ notations are as $n \to \infty$.

which can be easily expanded to the explicit formula:

$$\Pr[K_n = k] = \alpha_{n-k+1} \prod_{i=n-k+2}^{n} (1 - \alpha_i) \ .^{19} \tag{3.10}$$

Formula (3.10) can be interpreted in such the way: probabilistic snapshot was taken at the time step $n - k + 1$ and after this moment, there were $k - 1$ consecutive time steps without a substitution in a reservoir. By omitting the first event (i.e. the update of the sample at time step $n-k+1$), we may obtain the probability that probabilistic counter has at least value k:

$$\Pr[K_n \geqslant k] = \prod_{i=n-k+2}^{n} (1 - \alpha_i) \ , \tag{3.11}$$

where $n \in \mathbb{N}$ and $k \in [n]$. For our convenience, from now until the end of this section, we will denote $\Pr[K_n = k]$ by ${p_k}^{(n)}$.

The next theorem shows that the sequence $\mathbb{E}[K_n]$ satisfies simple linear first order difference equation:

Theorem 3.3.1. *Let $(\alpha_n)_n$ be a sequence of real numbers from the interval $[0, 1]$ with a starting condition $\alpha_1 = 1$. Moreover, let $(K_n)_n$ be a stochastic process which satisfies $\Pr[K_n = 1] = \alpha_n$ and (3.9) for any $n \in \mathbb{N}$ and $k \in [n]$. Then $\mathbb{E}[K_1] = 1$ and*

$$\mathbb{E}[K_{n+1}] = 1 + (1 - \alpha_{n+1})\mathbb{E}[K_n] \ .$$

Proof. Let $\beta_n = 1 - \alpha_n$. The first part of the proof is a straightforward consequence of the constraint $\alpha_1 = 1$ and the second expression follows directly from the recurrence (3.9):

$$\begin{aligned}
\mathbb{E}[K_{n+1}] &= \sum_{k=1}^{n+1} k p_k^{(n+1)} = \alpha_n + \sum_{k=2}^{n+1} k p_k^{(n+1)} \overset{(3.9)}{=} \alpha_n + \sum_{k=2}^{n+1} k(1 - \alpha_{n+1}) p_{k-1}^{(n)} \\
&= \alpha_n + \beta_{n+1} \sum_{k=1}^{n} (k+1) p_k^{(n)} = \alpha_n + \beta_{n+1} \sum_{k=1}^{n} k p_k^{(n)} + \beta_{n+1} \sum_{k=1}^{n} p_k^{(n)} \\
&= \alpha_n + \beta_{n+1} \sum_{k=1}^{n} k p_k^{(n)} + \beta_{n+1} = 1 + (1 - \alpha_{n+1})\mathbb{E}[K_n] \ .
\end{aligned}$$

□

Sometimes we would also like to utilize the foregoing random variable: $L_n := (n+1) - K_n$. It controls an index of a presently sampled data item. From formula (3.10), we immediately get

$$\Pr[L_n = k] = \alpha_k \prod_{i=0}^{n-k-1} (1 - \alpha_{n-i}) \ . \tag{3.12}$$

[19] Here $n \in \mathbb{N}$ and $k \in [n]$.

Lemma 11. *Monotonicity Let* $(\alpha_n)_n$ *and* $(\alpha'_n)_n$ *be update sequences of* $(K_n)_n$ *and* $(K'_n)_n$ *respectively. If* $(\forall\ n \in \mathbb{N})\quad 0 \leqslant \alpha_n \leqslant \alpha'_n \leqslant 1$, *then* $(\forall\ n \in \mathbb{N})\quad \mathbb{E}[K_n] \geqslant \mathbb{E}[K'_n]$.

Proof. The proof is a simple induction. Theorem 3.3.1 guarantees $\mathbb{E}[K_1] = \mathbb{E}[K'_1 = 1]$. Now, assume that $\mathbb{E}[K_n] \geqslant \mathbb{E}[K'_n]$ for some $n \in \mathbb{N}$. Then, by Theorem 3.3.1 once again,

$$\mathbb{E}[K_{n+1}] = 1 + (1 - \alpha_{n+1})\mathbb{E}[K_n] \geqslant 1 + (1 - \alpha'_{n+1})\mathbb{E}[K'_n] = \mathbb{E}\big[K'_{n+1}\big]\ .$$

□

3.3.4 Fixed value

We start our considerations with a case of $\alpha = 0$ and $g < 1$. This way we get simply $\alpha_n \equiv g$ for each $n > 1$ (we naturally exclude $n = 1$, since $\alpha_1 = 1$). This arrangement allows to provide a closed formula for $\mathbb{E}[K_n]$. Namely, from equation (3.10), we immediately get

$$\Pr[K_n = k] = \begin{cases} g(1-g)^{k-1} & : 1 \leqslant k < n \\ (1-g)^{n-1} & : k = n \end{cases}\ .$$

From the above formula, we deduce that:

1. Variables K_n have R-shifted version of Geo $(g, n-1)$ distribution, for $n \in \mathbb{N}$,
2. The sequence $(K_n)_n$ of random variables converges in distribution to the R-shifted version of Geo (p) distribution,
3. $\mathbb{E}[K_n] = \frac{1}{g}\left(1 - (1-g)^n\right)$, for each $n \in \mathbb{N}$.

Therefore the probabilistic counter K_n in this case may be used for controlling a behaviour of a stream at a vicinity of the $\left\lfloor \frac{1}{g} \right\rceil$-th time step apart from the present moment. In other words, we expect, that after the n-th time step (where $\frac{1}{g} \ll n$),[20] we sample contains one of the neighbours of the data item $S_{n+1-\lfloor \frac{1}{g} \rceil}$ from the stream.

3.3.5 Sublinear Case

In this part we consider the case when $\alpha_n = \min\left(1, \frac{g}{n^\alpha}\right)$ for some fixed $g > 0$ and $\alpha \in (0, 1)$. We show that the normalized random variable $\frac{K_n}{n^\alpha}$ converges in distribution to the exponential Exp (g) distribution.

[20] Obviously, when $n \approx \frac{1}{g}$, then $\mathbb{E}[K_n] \approx \frac{1-e^{-1}}{g}$. Hence the condition $\frac{1}{g} \ll n$ is quite crucial.

Theorem 3.3.2. *Let $g > 0$, $\alpha \in (0,1)$, $\alpha_n = \min(1, \frac{g}{n^\alpha})$ and $x > 0$. Then*

$$\lim_{n\to\infty} \Pr\left[\frac{K_n}{n^\alpha} \leqslant x\right] = 1 - e^{-gx} \ .$$

Proof. Let $k := \lfloor n^\alpha x \rfloor$. Using formula (3.11) we obtain

$$\Pr\left[\frac{K_n}{n^\alpha} > x\right] = \Pr[K_n > n^\alpha x] = \Pr[K_n > k] = \prod_{i=n-k+1}^{n} \left(1 - \frac{g}{i^\alpha}\right)$$

(for sufficiently large n).[21] Therefore

$$\Pr\left[\frac{K_n}{n^\alpha} > x\right] < \left(1 - \frac{g}{n^\alpha}\right)^k \leqslant \left(1 - \frac{g}{n^\alpha}\right)^{n^\alpha x - 1} \tag{3.13}$$

and

$$\begin{aligned}\Pr\left[\frac{K_n}{n^\alpha} > x\right] &> \left(1 - \frac{g}{(n-k+1)^\alpha}\right)^k > \left(1 - \frac{g}{(n - \lfloor n^\alpha x \rfloor)^\alpha}\right)^{n^\alpha x} \\ &= \left(1 - \frac{g}{n^\alpha \left(1 - \frac{\lfloor n^\alpha x \rfloor}{n}\right)^\alpha}\right)^{n^\alpha x} ,\end{aligned}$$

so we see that both bounds on $\Pr\left[\frac{K_n}{n^\alpha} > x\right]$ converge to the same limit e^{-gx} when n tends to infinity. □

We conclude from Theorem 3.3.2, that $\left(\frac{K_n}{n^\alpha}\right)_n$ converges in distribution to $X \sim \operatorname{Exp}(g)$.

Theorem 3.3.3. *If $g > 0$, $\alpha \in (0,1)$ and $\alpha_n = \min\left(1, \frac{g}{n^\alpha}\right)$, then*

$$\lim_{n\to\infty} \mathbb{E}\left[\frac{K_n}{n^\alpha}\right] = \frac{1}{g} \ .$$

Proof. From Theorem 3.3.2 and Fatou's Lemma 1 we get $\frac{1}{g} \leqslant \liminf_{n\to\infty} \mathbb{E}\left[\frac{K_n}{n^\alpha}\right]$.[22] On the other hand, inequality (3.13) bears (for sufficiently large $n > g^{\alpha^{-1}}$):

$$\begin{aligned}\lim_{n\to\infty} \mathbb{E}\left[\frac{K_n}{n^\alpha}\right] &= \lim_{n\to\infty} \int_0^\infty \Pr\left[\frac{K_n}{n^\alpha} > t\right] dt \leqslant \lim_{n\to\infty} \int_0^\infty \left(1 - \frac{g}{n^\alpha}\right)^{n^\alpha t - 1} dt \\ &= \lim_{n\to\infty} -\frac{1}{(n^\alpha - g)\ln(1 - \frac{g}{n^\alpha})} = \frac{1}{g} \ .\end{aligned}$$

□

Corollary 3. *If $g > 0$ and $\alpha_n = \min\left(1, \frac{g}{n^\alpha}\right)$ then $\mathbb{E}[K_n] = \frac{n^\alpha}{g} + \mathrm{o}(n^\alpha)$, as $n \to \infty$.*

[21] For n such that $(n - \lfloor n^\alpha x \rfloor + 1)^\alpha > g$. It worth to mention that this condition depends on x, so this assumption is satisfied earlier for smaller x.

[22] Recall that if $E \sim \operatorname{Exp}(g)$, then $\mathbb{E}[E] = \frac{1}{g}$.

Remark We know a more precise results in some special cases. For example, if $\alpha_n = \frac{1}{\sqrt{n}}$ then $\mathbb{E}[K_n] = \sqrt{n} - \frac{1}{2} + \frac{1}{2\sqrt{n}} + \frac{1}{8n} + \mathrm{O}\left(n^{-\frac{3}{2}}\right)$, as $n \to \infty$. We will take a closer look at such cases in Appendix D.

3.3.6 Linear Case

In this section we consider the case of $\alpha = 1$, i.e. when $\alpha_n = \min\left(1, \frac{g}{n}\right)$ for some fixed $g > 0$. We show that the normalized random variable $\frac{K_n}{n}$ converges in distribution to the $\mathrm{B}(1, g)$ distribution.

Theorem 3.3.4. *Let $g > 0$, $\alpha_n = \min\left(1, \frac{g}{n}\right)$ and $x \in (0, 1)$. Then*

$$\lim_{n\to\infty} \Pr\left[\frac{K_n}{n} \leqslant x\right] = 1 - (1-x)^g \ .$$

Proof. Analogously to the reasoning from the proof of Theorem 3.3.2, let $k := \lfloor nx \rfloor$. Then we attain

$$\Pr\left[\frac{K_n}{n} > x\right] = \Pr[K_n > nx] = \Pr[K_n > k] = \prod_{i=n-k+1}^{n} (1 - \alpha_i) \ .$$

Therefore, for each $x \in (0, 1)$, there exists sufficiently large n, that we have

$$\Pr\left[\frac{K_n}{n} > x\right] = \prod_{i=n-k+1}^{n} \left(1 - \frac{g}{i}\right) \ .$$

Hence

$$\begin{aligned}
\ln\left(\Pr\left[\frac{K_n}{n} > x\right]\right) &= \sum_{i=n-k+1}^{n} \ln\left(1 - \frac{g}{i}\right) \overset{(1.5)}{=} -\sum_{i=n-k+1}^{n} \sum_{a=1}^{\infty} \frac{1}{a} \left(\frac{g}{i}\right)^a \\
&= -g \sum_{i=n-k+1}^{n} \frac{1}{i} - \sum_{a=2}^{\infty} \frac{g^a}{a} \sum_{i=n-k+1}^{n} \frac{1}{i^a} \ .
\end{aligned}$$

Notice that $\sum_{i=n-k+1}^{n} \frac{1}{i} = H_n - H_{n-k}$ (see Section 1.2.9). Thence according to Fact 1.2.11, we obtain

$$\begin{aligned}
-g(H_n - H_{n-k}) &= -g\left(\ln(n) - \ln(n-k) + \mathrm{O}\left(\frac{1}{n}\right) + \mathrm{O}\left(\frac{1}{n - \lfloor nx \rfloor}\right)\right) \\
&= \ln\left(\frac{n - \lfloor nx \rfloor}{n}\right)^g + \mathrm{O}\left(\frac{1}{n}\right) \ ,
\end{aligned} \tag{3.14}$$

as $n \to \infty$. Therefore $\lim_{n\to\infty} (-g(H_n - H_{n-\lfloor nx \rfloor})) = \ln((1-x)^g)$. Moreover, let

$$A_{n,k} := \sum_{a=2}^{\infty} \frac{g^a}{a} \sum_{i=n-k+1}^{n} \frac{1}{i^a} \left(= \sum_{a=2}^{\infty} \frac{g^a}{a} \left(H_n^{(a)} - H_{n-k+1}^{(a)}\right)\right) \ .$$

Note that the summands of beneath are non-negative, hence when $n-\lfloor nx\rfloor > 2g$, according to Tonneli's Theorem 1.4.8 we obtain

$$0 < A_{n,k} < \sum_{a=2}^{\infty} g^a \sum_{i=n-k+1}^{\infty} \frac{1}{i^a} = \sum_{i=n-k+1}^{\infty} \sum_{a\geqslant 2} \frac{g^a}{i^a}$$
$$= \sum_{i=n-k+1}^{\infty} \frac{g^2}{i^2} \frac{1}{1-\frac{g}{i}} < 2g^2 \sum_{i=n-k+1}^{\infty} \frac{1}{i^2} \ .$$

Consequently we get

$$A_{n,k} < 2g^2 \sum_{i=n-k+1}^{\infty} \frac{1}{i(i-1)} = \frac{2g^2}{n-k} = \frac{2g^2}{n-\lfloor nx\rfloor} = \mathrm{O}\left(\frac{1}{n}\right) \tag{3.15}$$

as $n \to \infty$. Thus from (3.14) and (3.15) we attain

$$\ln\left(\Pr\left[\frac{K_n}{n} > x\right]\right) = \ln\left(1-\frac{\lfloor nx\rfloor}{n}\right)^g + \mathrm{O}\left(\frac{1}{n}\right),$$

as $n \to \infty$, and consequently

$$\Pr\left[\frac{K_n}{n} > x\right] = \left(1-\frac{\lfloor nx\rfloor}{n}\right)^g e^{\mathrm{O}\left(\frac{1}{n}\right)} \overset{(1.4)}{=} \left(1-\frac{\lfloor nx\rfloor}{n}\right)^g + \mathrm{O}\left(\frac{1}{n}\right) \ ,$$

as $n \to \infty$, so $\lim_{n\to\infty} \Pr\left[\frac{K_n}{n} > x\right] = (1-x)^g$. □

Corollary 4. *If $g > 0$ and $\alpha_n = \min\left(1, \frac{g}{n}\right)$ then $\lim_{n\to\infty} \mathbb{E}\left[\frac{K_n}{n}\right] = \frac{1}{g+1}$.*

Proof. Notice that $\frac{K_n}{n} \leqslant 1$, hence the sequence $\left(\frac{K_n}{n}\right)_{n\in\mathbb{N}}$ is bounded, therefore the convergence in distribution of the sequence $\left(\frac{K_n}{n}\right)_{n\in\mathbb{N}}$ to a random variable Y with $\mathrm{B}(1,g)$ distribution (Theorem 3.3.4) implies the convergence of moments (see e.g. [14]), hence

$$\lim_{n\to\infty} \mathbb{E}\left[\frac{K_n}{n}\right] = \int_0^1 x \frac{d}{dx}\left(1-(1-x)^g\right) dx = \frac{1}{1+g} \ .$$

□

Corollary 5. *If $g > 0$ and $\alpha_n = \min\left(1, \frac{g}{n}\right)$, then $\mathbb{E}[K_n] = \frac{n}{g+1} + \mathrm{o}(n)$, as $n \to \infty$.*

Remark A slightly more complicated calculus shows that in the case considered in this section, one can obtain $\mathbb{E}[K_n] = \frac{n+1}{g+1} + \mathrm{O}\left(n^{-g}\right)$, as $n \to \infty$. We will also have a closer look at this particular case in Appendix D.

3.3.7 Subquadratic Case

In this subsection we consider the case when $\alpha_n = \min\left(1, \frac{g}{n^\alpha}\right)$ for some fixed $g > 0$ and $\alpha \in (1,2)$. For $\alpha > 1$ the behaviour of K_n slightly changes, so as we announced before, we are going to use the auxiliary random variable $L_n = n + 1 - K_n$ instead. Let us define $g_0 = \left\lceil g^{\frac{1}{\alpha}} \right\rceil - 1$. Realize that $\alpha_k = \frac{g}{k^\alpha}$ for $k \geqslant g_0 + 1$. Thus we briefly see that $\Pr[L_n = k] = 0$, whenever $k \leqslant g_0$, and otherwise, according to formula (3.12), we get

$$\Pr[L_n = k] = \frac{g}{k^\alpha} \prod_{i=0}^{n-k-1} \left(1 - \frac{g}{(n-i)^\alpha}\right) .$$

In foregoing calculations let $k > g_0$. Let us commence with a consideration about a lower bound, obtained with a use of standard Weierstrass' Product inequality (Theorem 1.4.2):

$$\Pr[L_n = k] \geqslant \frac{g}{k^\alpha} \left(1 - \sum_{i=k+1}^{n} \frac{g}{i^\alpha}\right) = \frac{g}{k^\alpha} \left(1 - g\left({H_n}^{(\alpha)} - {H_k}^{(\alpha)}\right)\right) . \quad (3.16)$$

To find the upper limitation, we apply more precise extension of Weierstrass' Product Inequality (Theorem 1.4.2):

$$\begin{aligned} \Pr[L_n = k] &\overset{\text{WPI}}{\leqslant} \frac{g}{k^\alpha} \left(1 - \sum_{i=k+1}^{n} \frac{g}{i^\alpha} + \sum_{i=k+1}^{n-1} \sum_{j=i+1}^{n} \frac{g^2}{(ij)^\alpha}\right) \\ &\leqslant \frac{g}{k^\alpha} \left(1 - \sum_{i=k+1}^{n} \frac{g}{i^\alpha} + \frac{1}{2} \sum_{i=k+1}^{n} \sum_{j=k+1}^{n} \frac{g^2}{(ij)^\alpha}\right) . \end{aligned} \quad (3.17)$$

Hence, by repeating the same trick as in (3.16) we attain:

$$\Pr[L_n = k] \leqslant \frac{g}{k^\alpha} \left(1 - g\left({H_n}^{(\alpha)} - {H_k}^{(\alpha)}\right) + \frac{g^2}{2}\left({H_n}^{(\alpha)} - {H_k}^{(\alpha)}\right)^2\right) .$$

We would like to find close confinements for an expected value of L_n.
In the following consideration of a lower limitation of $\mathbb{E}[L_n]$ (which is easier to analyze), we extensively abuse Fact 1.2.10 and Fact 1.2.11:

$$\begin{aligned} \mathbb{E}[L_n] &\overset{(3.16)}{\geqslant} \sum_{k=g_0+1}^{n} \frac{g}{k^{\alpha-1}} \left(1 - g\left({H_n}^{(\alpha)} - {H_k}^{(\alpha)}\right)\right) \\ &= g\left({H_n}^{(\alpha-1)} - {H_{g_0}}^{(\alpha-1)}\right) - g^2 \sum_{k=g_0+1}^{n} \frac{\frac{k^{1-\alpha} - n^{1-\alpha}}{\alpha-1} + f_\alpha(k)}{k^{\alpha-1}} \\ &= g\frac{n^{2-\alpha}}{2-\alpha} - g^2 \frac{\left({H_n}^{(2\alpha-2)} - {H_{g_0}}^{(2\alpha-2)}\right) - n^{1-\alpha}\left({H_n}^{(\alpha-1)} - {H_{g_0}}^{(\alpha-1)}\right)}{\alpha-1} + \mathrm{O}(1) \\ &= \frac{gn^{2-\alpha}}{2-\alpha} - g^2 \left[-\frac{n^{3-2\alpha}}{(\alpha-1)(2-\alpha)} + \frac{\left({H_n}^{(2\alpha-2)} - {H_{g_0}}^{(2\alpha-2)}\right)}{\alpha-1}\right] + \mathrm{O}(1) = \ldots \end{aligned}$$

$$\ldots = \begin{cases} \frac{gn^{2-\alpha}}{2-\alpha} - \frac{g^2 n^{3-2\alpha}}{(3-2\alpha)(\alpha-1)} + \frac{g^2 n^{3-2\alpha}}{(\alpha-1)(2-\alpha)} + \mathrm{O}(1) & ; \text{ for } \alpha < \frac{3}{2}, \\ \frac{gn^{2-\alpha}}{2-\alpha} - \frac{g^2 H_n}{\alpha-1} + \mathrm{O}(1) & ; \text{ for } \alpha = \frac{3}{2}, \\ \frac{gn^{2-\alpha}}{2-\alpha} + \mathrm{O}(1) & ; \text{ for } \alpha > \frac{3}{2}, \end{cases}$$

$$= \begin{cases} \frac{gn^{2-\alpha}}{2-\alpha} - \frac{g^2 n^{3-2\alpha}}{(3-2\alpha)(2-\alpha)} + \mathrm{O}(1) & ; \text{ for } 1 < \alpha < \frac{3}{2}, \\ \frac{gn^{2-\alpha}}{2-\alpha} - \frac{g^2 H_n}{\alpha-1} + \mathrm{O}(1) & ; \text{ for } \alpha = \frac{3}{2}, \\ \frac{gn^{2-\alpha}}{2-\alpha} + \mathrm{O}(1) & ; \text{ for } 2 > \alpha > \frac{3}{2}, \end{cases}$$

as $n \to \infty$, where $|f_\alpha(k)| < k^{-\alpha}$ (see Fact 1.2.10).[23]
The second bound can be attained in similar way:

$$\mathbb{E}[L_n] \leqslant \begin{cases} \frac{gn^{2-a}}{2-a} - \frac{g^2 n^{3-2a}}{(3-2a)(2-a)} + \frac{g^2 n^{4-3a}}{(4-3a)(3-2a)(2-a)} + \mathrm{O}(1) & ; \text{ for } 1 < a < \frac{4}{3}, \\ \frac{gn^{2-a}}{2-a} - \frac{g^2 n^{3-2a}}{(3-2a)(2-a)} + \frac{g^3 H_n}{2(a-1)^2} + \mathrm{O}(1) & ; \text{ for } a = \frac{4}{3}, \\ \frac{gn^{2-a}}{2-a} - \frac{g^2 n^{3-2a}}{(3-2a)(2-a)} + \mathrm{O}(1) & ; \text{ for } \frac{4}{3} < a < \frac{3}{2}, \\ \frac{gn^{2-a}}{2-a} - \frac{g^2 H_n}{a-1} + \mathrm{O}(1) & ; \text{ for } a = \frac{3}{2}, \\ \frac{gn^{2-a}}{2-a} + \mathrm{O}(1) & ; \text{ for } 2 > a > \frac{3}{2}, \end{cases}$$

as $n \to \infty$.

Let us summarize all the cases:

Corollary 6. *If $g > 0$ is fixed, $\alpha \in (1, 2)$ and $\alpha_n = \min\left(1, \frac{g}{n^\alpha}\right)$, then*

$$\mathbb{E}[L_n] = \begin{cases} \frac{gn^{2-\alpha}}{2-\alpha} - \frac{g^2 n^{3-2\alpha}}{(3-2\alpha)(2-\alpha)} + \mathrm{O}\left(\max\left(\ln n, n^{4-3\alpha}\right)\right) & ; \text{ for } \alpha < \frac{3}{2}, \\ \frac{gn^{2-\alpha}}{2-\alpha} - \frac{g^2 H_n}{\alpha-1} + \mathrm{O}(1) & ; \text{ for } \alpha = \frac{3}{2}, \\ \frac{gn^{2-\alpha}}{2-\alpha} + \mathrm{O}(1) & ; \text{ for } \alpha > \frac{3}{2}, \end{cases}$$

as $n \to \infty$.

Remark One can use more precise generalization of Weierstrass' Product Inequality (Theorem 1.4.3) in order to provide better approximations of $\mathbb{E}[L_n]$, however, with every additional expression, the calculations get complicated significantly.

3.3.8 Quadratic Case

In this subsection we consider the case of $\alpha = 2$, i.e. when $\alpha_n = \min\left(1, \frac{g}{n^2}\right)$ for some fixed $g > 0$. For computational purpose let introduce some auxiliary symbols: $g_0 := \sqrt{g}$ and $h := \lceil g_0 \rceil$. Again we will focus on the random variable $L_n = n + 1 - K_n$. Our main goal is to achieve an approximation for its expected value. It is easy to realize that $\Pr[L_n \geqslant k] = 1$, when $k < h$. In the other cases, we use (3.11) to reach $\Pr[L_n \geqslant k] = 1 - \prod_{i=k}^{n} \left(1 - \frac{g}{i^2}\right)$.

[23] Let us note that $\sum_{k=1}^{\infty} \frac{f_\alpha(k)}{k^{\alpha-1}} \leqslant \zeta(2\alpha - 1)$, which is finite for $\alpha > 1$.

Theorem 3.3.5. *Let $g > 0$ and $\alpha_n = \min\left(1, \frac{g}{n^2}\right)$. Then $\mathbb{E}[L_n] = g \ln n + \mathrm{O}(1)$, as $n \to \infty$.*

Proof. Commence with some simple facts. The first one uses a discrete version of Fact 1.3.1:

$$\mathbb{E}[L_n] = \sum_{k=1}^{n} \Pr[L_n \geqslant k] = h - 1 + \sum_{k=h}^{n} \left(1 - \frac{g}{k^2}\right) \ . \tag{3.18}$$

The next applies inequality (1.3):

$$\sum_{i=k}^{n} i^{-2} \leqslant \int_{k-1}^{n} x^{-2} dx = -(n^{-1} - (k-1)^{-1}) \tag{3.19}$$

Now, we are ready to find an upper bound for $\Pr[L_n \geqslant k]$ (for $k \geqslant h$) using standard Weierstrass' Product Inequality (Theorem 1.4.2) and (3.19):

$$\Pr[L_n \geqslant k] \leqslant 1 - \prod_{i=k}^{n} \left(1 - \frac{g}{i^2}\right) \leqslant 1 - \left(1 - \sum_{i=k}^{n} \frac{g}{i^2}\right) \leqslant g\left(\frac{1}{k-1} - \frac{1}{n}\right) \ .$$

From the above formula and (3.18), we obtain:

$$\begin{aligned} \mathbb{E}[L_n] &\leqslant h - 1 + g \sum_{k=h}^{n} \left(\frac{1}{k-1} - \frac{1}{n}\right) \\ &= g H_{n-1} - g H_{h-2} + h - 1 - g + \mathrm{O}\left(n^{-1}\right) = g \ln n + \mathrm{O}(1) \ , \end{aligned}$$

as $n \to \infty$.

In order to achieve the opposite bound on the expected value of L_n, we are going to utilize the formula (3.16):[24]

$$\begin{aligned} \mathbb{E}[L_n] &\overset{(3.16)}{\geqslant} \sum_{k=h}^{n} \frac{g}{k} \left(1 - g\left(H_n{}^{(2)} - H_k{}^{(2)}\right)\right) \\ &\overset{(3.19)}{\geqslant} g(H_n - H_{h-1}) - g^2 \sum_{k=h}^{n} \left(\frac{1}{(k-1)k} - \frac{1}{nk}\right) \\ &= g \ln n - g^2 \left(\frac{1}{h-1} - \frac{1}{n} - \frac{H_n - H_{h-1}}{n}\right) + \mathrm{O}(1) = g \ln n + \mathrm{O}(1) \ , \end{aligned}$$

as $n \to \infty$. □

Remark Let us mention that the first version of the proof of Theorem 3.3.5 utilized Exponential Integrals Ei and E_1 (see [1] for definitions and properties) and several additional formulas. However the presented proof is much simpler.

[24] Although this formula was provided, when we concerned the case $\alpha < 2$, it is still in use, when $\alpha = 2$.

Remark Using falling factorial (see Section 1.2.8 for definition) and the equation in the formula (3.18), we can write:

$$\mathbb{E}[L_n] = h^2 \frac{[n+h]_h}{[n]_h} \sum_{k=h}^{n} \frac{[k-1]_{h-1}}{[k+h]_h}$$

as $n \to \infty$.Let us highlight that the above sum can be attained by Gosper—Zeilberger algorithm ([53, 106, 110]). Nevertheless the precise calculation is quite complicated and will be omitted.

Remark The case $\alpha = 2$ may be recognized as some kind of exception, since the crucial term is related to H_n. However, one may notice that there exists a pattern for the expected value of L_n, which extrapolates from the case $1 < \alpha < 2$ to the one with $\alpha = 2$.

Corollary 7. *If $h \in \mathbb{N}$, $g = h^2$ and $\alpha_n = \min\left(1, \frac{g}{n}\right)$, then $\mathbb{E}[K_n] = n + 1 - gH_n + \mathrm{O}(1)$, as $n \to \infty$.*

3.3.9 Superquadratic Case

Finally, in this section we consider the case when $\alpha_n = \min\left(1, \frac{g}{n^\alpha}\right)$ for some fixed $g > 0$ and $\alpha > 2$. This case is the least useful in our purpose, but we concisely consider a case $a > 2$ for completeness. Again, we apply random variable $L_n = n + 1 - K_n$. The foregoing obvious fact is fulfilled:

$$\Pr[L_n = k] = \alpha_k \prod_{i=0}^{n-k-1} (1 - \alpha_{n-i}) \leqslant \alpha_k = \min\left(1, \frac{g}{k^\alpha}\right) .$$

Moreover, the foregoing formula provides an easy majorization of the expected value:

$$\mathbb{E}[L_n] \leqslant \sum_{k=1}^{\left\lfloor g^{\frac{1}{\alpha}} \right\rfloor} k + gH_{n,\alpha-1} < \frac{g^{\frac{1}{\alpha}} + 1}{2} g^{\frac{1}{\alpha}} + g\zeta(\alpha - 1) < \infty .$$

Corollary 8. *If $\alpha > 2$, $g > 0$ and $\alpha_n = \min\left(1, \frac{g}{n^\alpha}\right)$, then $\mathbb{E}[K_n] = n - \mathrm{O}(1)$.*

Remark L_n is a number of update of algorithm which saved the last snapshot. We should expect that our algorithm save only data from the very beginning. In this case, the algorithm is redundant, so we don't consider this case anymore. Nevertheless this case arises a quite challenging mathematical problem to study.

3.3.10 Applications

Here we enumerate some possible applications of our contribution from this Section 3.3:

- The solution may be utilized to store fixed number of snapshots from an observed movie of unknown length. For example, we may want to store short samples (a single frame or a fragment of the movie) taken at times close to 0, $\frac{1}{10}T$, $\frac{2}{10}T$ ... $\frac{9}{10}T$, T from a movie of length T.[25] In this case we should use linear PLU. We will take a closer look on this case later.
- We may need a sample of data from times close to n, $n-C$, $n-2\cdot C$,..., $n-k\cdot C$, where n is an index of current item, C is a fixed time length and k is a reasonably small natural number. For instance, when we are observing a sensor which measures temperature each second, it is not very likely for the temperature to change roughly. Moreover, we may be interested to know only the approximation of the lowest and the highest temperatures in each day. In such the situation, C should be tuned up to reflect 12 hours. This way we obtain the information of the extremal temperatures from the last $\frac{k+1}{2}$ days. When one wants to achieve more certain information, it is also possible to repeat independently the aforementioned procedure, but shifted in time by e.g. half an hour (i.e. the second run of the algorithm starts half an hour later). In order to handle this scenario, we need to apply independent versions of fixed PLU with $k+1$ different parameters.
- We may need to observe a sample from a stock market in such a way that the snapshots from the past should be less rare than snapshots from times close to the present. This can be done by utilizing sublinear PLU.

Linear sampling

Let us assume that we are observing a data stream and after reading n-th item we would like to have access to elements laying near points $\left\{\frac{k}{10}\cdot n : k\in[0:10]\right\}$. Of course, there is no problem with the element laying near $\frac{0}{n}\cdot n$ — it is sufficient to store the first element from the stream. As an element laying near $\frac{10}{10}\cdot n$ we may take the current item. So we must propose some mechanism for dealing with remaining 9 points.

Let us consider a series $K_n^1,\ldots,K_n^M$ of independent random variables generated by the control sequence $\alpha_n=\frac{1}{n}$. We know (see Section 3.3.6) that this is a sequence of independent random variables uniformly distributed in the set $[n]$. Let $X_n=\left\{K_n^1,\ldots,K_n^M\right\}$. Let us fix some $0<\varepsilon<\frac{1}{10}$. Let $I_k=\{a\in\mathbb{N}:\left|\frac{a}{n}-\frac{k}{10}\right|<\varepsilon\}$. Let us observe that $\Pr[I_k\cap X_n=\emptyset]\approx(1-2\varepsilon)^M$. This approximation is accurate for large n. So, for simplicity we shall assume that we have an equality. Therefore

$$\Pr\left[\bigvee_{k=1}^{9}(I_k\cap X_n=\emptyset)\right]\leqslant 9\cdot(1-2\varepsilon)^M\ .$$

The solution of inequality $9\cdot(1-2\varepsilon)^M\leqslant\eta$ is given by $M\geqslant\dfrac{\ln\left(\frac{\eta}{9}\right)}{\ln(1-2\varepsilon)}$. Putting in this formula $\varepsilon=\frac{1}{100}$ and $\eta=10^{-10}$ we get $M\geqslant 1248.5$. Therefore, if we

[25] This example justifies the name "probabilistic snapshot" per se.

take $M = 1250$ snapshots then

$$\Pr\left[\bigwedge_{k=1}^{9}(I_k \cap X_n \neq \emptyset)\right] > 1 - \frac{1}{10^{10}} \ .$$

Hence, with a very high probability for each $k \in [0:10]$ we are able to choose a point from the set X_n which approximates $\frac{kn}{10}$ with precision 1%.

Bitcoin capitalization

In Figure 3.5 we compared bitcoin cap data from [90] with experimental approximation by probabilistic snapshots concentrated at the end of the data stream (the plot has reversed time, so the latest data items are indicated by the lowest numbers in Figure 3.5, i.e. 1 refers to the date 2017-12-31 and 1439 – to 2013-09-30). The aforementioned data stream consists of 1439 records, from 2013-09-30 to 2017-12-31 (one record per day). Since this length is relatively short, we decided to use only 100 probabilistic snapshots with update sequence $\alpha_n = \frac{0.1}{\sqrt{n}}$. From Section 3.3.5 we know that in this case the expected value of time steps that elapsed since the last update of the snapshot was taken is close to $10\sqrt{1439} \approx 379$.

We added to generated probabilistic snapshots to points: the first one, and the last one and further completed the plot via linear interpolation (we connected the points given by the snapshots). Note that despite the large fluctuations in the bitcoin market at the end of 2017, only 100 snapshots makes it possible to reproduce the main trend of this parameter of the bitcoin market quite faithfully.[26]

[26] Let us recall, that the main trend of bitcoin cap has dramatically changed in 2017, what makes it more difficult to approximate. This is why we consider here this old dataset.

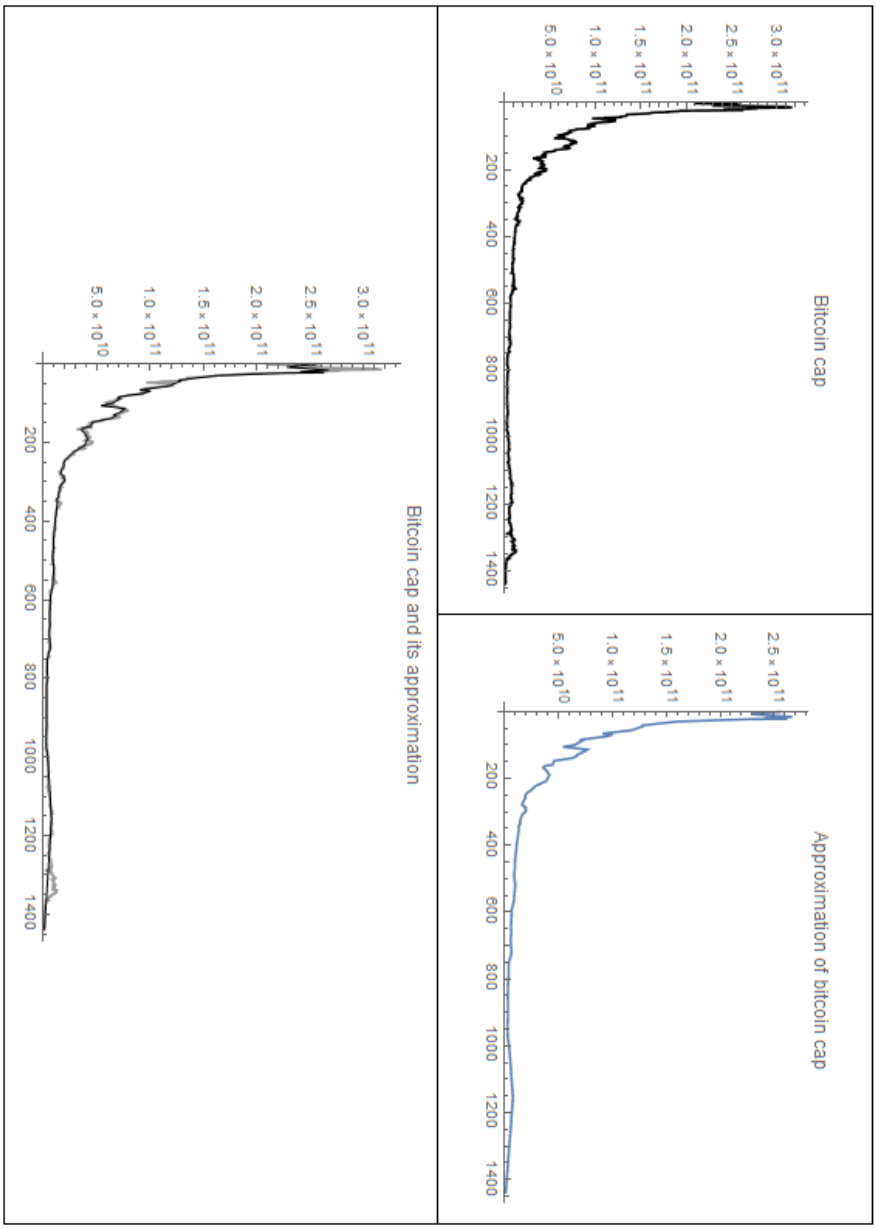

Figure 3.5: Precision of bitcoin cap approximation by 100 independent snapshots with update sequence $\alpha_n = \frac{0.1}{\sqrt{n}}$.

Chapter 4

Inherent Privacy of Probabilistic Counters

4.1 Introduction

Since Big Data related topics widely developed in recent years, solutions which focus on saving memory resources have become very popular. We are going to consider probabilistic counters, which are well known tools often used for space-efficient cardinality estimation of dynamically counted events. We would like to indicate occurrence of n events using very small (significantly less than $\lg n$) number of bits. We assume that n is unknown in advance and may change. Clearly a simple information-theoretic argument convinces us that is not feasible if we demand an exact representation of the number of events. Nevertheless, there are some very efficient solutions via probabilistic counters that require only $\Theta(\lg \lg n)$ bits and guarantee sufficient accuracy for a wide range of applications. Probabilistic counters are known in the literature since the seminal Morris' paper [80] followed by its thorough mathematical analysis by Flajolet in [44]. They are used as building blocks in many space-efficient algorithms in the field of data mining or distributed data aggregation in networks (like [8] or [20]) just to mention a few.

In this chapter we investigate probabilistic counters from the perspective of privacy protection. The analysis is based on differential privacy notion, which is commonly considered as the only state-of-the-art approach. The differential privacy has the undeniable advantage of being mathematically rigorous and formally provable in contrary to previous anonymity-derived privacy definitions. This approach to privacy-preserving protocols can be used to give formal guarantee for privacy which is resilient to any form of post-processing. For a survey about differential privacy properties see [40] and references therein. The analysis of protocols based on the differential privacy is usually technically complex, but by using it we are immune against e.g. linkage attacks (see for example [84, 85]).

Informally, the idea behind differential privacy is as follows: for two "neighbouring" scenarios that differ only in participation of a single *user* (*individual*), a differentially private mechanism should provide a response chosen from very similar distributions. Roughly speaking, the differential privacy is described by two parameters: ε – which controls the similarity of probabilities of common events – and δ – which is related to the probability of uncommon events. The smaller the parameters, the better from the privacy point of view. In effect, judging by the output of the mechanism one cannot say if a given individual was taken into account for producing a given output. Intuitively, probabilistic counters should provide high level of differential privacy, since statistically many various number of events should be "squeezed" into a small space of possible output results. However, it turned out, that providing a precise, formal analysis of differential privacy parameters of probabilistic counters is surprisingly difficult and needs a very careful approach.

Our primary motivation is to find possibly accurate privacy parameters of two most fundamental probabilistic counter protocols, namely Morris Counter from [80] and MaxGeo Counter from [99]. Note that the second one is used for yet another popular algorithm — HyperLogLog [45].[1] One may realize that these two counters are relatively old, but they, and their modifications, are extensively used until these days. Morris counter is often used in Big Data solutions, for example for measurement of network's capabilities [41]. We mention the most crucial examples of refinements of HyperLogLog algorithm in Section 4.7. We claim that a high precision analysis in the case of probabilistic counters is particularly crucial. This is due to the fact that even a mechanism with a very good privacy parameters may cause a serious privacy breach when used multiple times (see e.g. [40]). Clearly, probabilistic counters in realistic scenarios may be used as very fundamental primitives and subroutines in more complex protocols, since the differential privacy property is immune to post-processing.

We also show that they can be used safely without any additional randomization, even in a very demanding settings. It is commonly known that no deterministic algorithm can provide non-trivial differential privacy. Probabilistic counters, however, possess inherent randomness, which achieves desired privacy parameters. In other words, one can say that probabilistic counters are safe by design and we do not need any additional privacy oriented methods. In particular, a fact that existing and working implementations do not need to be changed if we start demanding a provable privacy of a system.

Finally we demonstrate how our results can be used for constructing a data aggregation protocol based on probabilistic counters that can be used in some specific distributed scenarios and compare the properties of counter-based solutions and a standard Laplace method.

[1] Let us point out, that we have already considered MaxGeo variables in Chapter 2.

4.2 Differential Privacy Preliminaries

In this section we briefly recall *differential privacy*. For more details see for example [40]. We assume that there exists a trusted *curator* who holds, or securely obtains, the data of *individuals* in a (possibly distributed) *database* x.Every row consists of the data of some individual. By $\mathcal{X}$ we denote a space of all possible rows. The goal is to protect the data of every single individual, even if all users except one collude with an *adversary* to breach privacy of this single, uncorrupted user. On the other hand, the curator is responsible for producing a *release* — a possibly accurate response to a requested *query*. The curator may be solicited be either one of the users or some external entity to answer the query. Release is then publicly posted, hence it is allowed to perform statistical analysis on it. The differential privacy is by design resilient to post-processing attacks, so even if the adversary obtains the public release, he will not be able to infer anything about specific individuals participating in that release.

For simplicity we interpret databases as theirs histograms in $\mathbb{N}_0^{\mathrm{card}(\mathcal{X})}$, so we may just focus on unique rows and the numbers of theirs occurrences.

Definition 4.2.1. *A distance between two databases x and y is defined as ℓ_1 norm of $x - y$, where x_i and y_i are number of occurrences of an item (an individual) i in databases x and y accordingly, i.e.*

$$\|x - y\|_1 = \sum_{i \in \mathcal{X}} |x_i - y_i|.$$

Then $\|x\|_1$ measures a size of the database x.

A *privacy mechanism* is a randomized algorithm used by the curator that takes as input a database, and produces the output (the release) using randomization.

Definition 4.2.2 (Differential Privacy (formulation from [40]))**.** *A randomized algorithm $\mathcal{M}$ with domain $\mathbb{N}^{\mathrm{card}(\mathcal{X})}$ is (ε, δ)-differentially private, if for all $\mathcal{S} \subseteq \mathrm{Range}(\mathcal{M})$ and for all $x, y \in \mathbb{N}^{\mathrm{card}(\mathcal{X})}$ such that $\|x - y\|_1 \leqslant 1$ the following condition is satisfied:*

$$\Pr[\mathcal{M}(x) \in \mathcal{S}] \leqslant e^{\varepsilon} \cdot \Pr[\mathcal{M}(y) \in \mathcal{S}] + \delta,$$

where the probability space is over the outcomes of the mechanism $\mathcal{M}$.

In case, when $\delta = 0$, $\mathcal{M}$ is called (ε)-differentially private mechanism.

An intuition of (ε, δ)-DP is as follows: if we choose two consecutive databases (that differ exactly on one record (row)), then the mechanism is very likely to return indistinguishable values. In other words, it preserves privacy with high probability, but it is admissible for mechanism to be out of control with some negligible probability δ.

Example 10 (Laplace noise)**.** *In central model, a standard, widely used mechanism with (ε)-differential privacy property is, so called, Laplace noise. Let $c(x)$*

be a number of rows in x, which satisfy an arbitrary property. Imagine that an aggregating mechanism is defined as follows: $\mathcal{M}(x) = c(x) + \mathcal{L}(\varepsilon^{-1})$. Then $\mathcal{M}$ is (ε)-differentially private (for more precise properties of Laplace noise method see [40]).

4.3 Probabilistic Counters

For a general description of probabilistic counters, see Section 1.3.4. Throughout this chapter, we denote probabilistic counters as $M = (M_n)_{n\in\mathbb{N}_0}$ and interpret them in terms of a mechanism, defined on space $\Omega = \mathcal{X}$ of all possible inputs.

Let us recall that each increase of the data source which is being counted by the probabilistic counter is called the incrementation request ('1'). Due to randomized nature of probabilistic counters it may increase the value of the counter, but not necessarily. For the sake of generality, we also assume that the counter can get as an input a fake request ('0'), and in such case it simply does nothing. This is useful for some real-life scenarios, e.g. data aggregation (see Section 4.6). We would like to emphasize that obviously only incrementation requests are impacting the final result of the counter, hence hereinafter n indicates the number of incrementation requests, when we are considering M_n. We would like to show that if we reveal the final result it does not expose any sensitive data about any single record. Moreover, note that if x and y differs only by one '0' input, then $\Pr[M(x) \in \mathcal{S}] = \Pr[M_n \in \mathcal{S}] = \Pr[M(y) \in \mathcal{S}]$, where n is the number of incrementation requests both in x and y. See that then the condition in Definition 4.2.2 is trivially fulfilled.

Fact 4.3.1. *Let M be a probabilistic counter with a discrete support A. Moreover assume that, for all $n, m \geqslant 1$ such that $|n - m| \leqslant 1$, there exists such $S_n \subset A$ that for all $s \in S_n$*

$$\Pr[M_n = s] \leqslant \exp(\varepsilon) \cdot \Pr[M_m = s] \tag{4.1}$$

and

$$\Pr[M_n \notin S_n] \leqslant \delta \ . \tag{4.2}$$

Then M is (ε, δ)-DP.

Note that, for our setting, Fact 4.3.1 is fully compatible with the intuition of regular differential privacy (Definition 4.2.2). Indeed, Fact 4.3.1 can be easily derived from the observation that any set $B \subset A$ is a disjoint union of $B \cap S_n$ and $B \cap S_n'$.

Let us emphasize that ε and δ in Fact 4.3.1 can be treated as functions of n parameter. Thence we are able to interpret $(\varepsilon(n), \delta(n))$-differential privacy of $(M_n)_n$ as the property of a mechanism which got exactly n incrementation request in its input. This approach let us to provide precise dependence of privacy parameters of the counter as the number of incrementation requests gets large.

4.3.1 Morris Counter

We begin with a short description of Morris Counter (originally referred to as an approximate counter [80, 44]). Let us fix $a > 1$. Algorithm 11 is a very simple adaptation of pseudo-code of Morris Counter from [80].

ALGORITHM 11: Morris Counter Mechanism

```
   procedure Init()
1  |  M = 1
   procedure Aggregate()                    // on incrementation request
1  |  if random() < a^{-M} then
2  |  |  M = M + 1
   procedure Get()
1  |  return M
```

Roughly speaking, we start with counter M set to 1 (procedure `Init`).[2] Each incoming incrementation request triggers a random event realized by *random()* (pseudo-random number generator), which draws a real number uniformly from the interval $(0, 1)$. This event increments the counter (line 1 of `Aggregate`) with probability a^{-M}.

Note that this approximate counting protocol can be easily distributed. Indeed, any entity who wants to increment the counter, only has to send the request to increment it. These requests can be queued on the server and resolved one after another.

Note that the standard Morris Counter algorithm consists of methods `Init` and `Aggregate`. However, in order to adapt it to differential privacy notion, we added `Get` method, which should be invoked by the curator when all the requests have been already read, in order to publish the aggregated counter.

A detailed description of the approximate counting method can be found in [80, 44]. Throughout this dissertation we examine only a standard Morris Counter, i.e. the one with the base $a = 2$.

Fact 4.3.2. *Morris Counter can be represented as the general counter according to Definition 1.3.1, for* $M_0 = 1$, $X(M_n) \sim \mathrm{Ber}\left(2^{-M_n}\right)$ *and* $f(x, y) = x + y$.

Morris Counter can be also defined recursively in the following way:

Definition 4.3.1. *Morris Counter is a Markov process* $(M_n)_{n\in\mathbb{N}_0}$ *which satisfies:*

- $\Pr[M_0 = 1] = 1$,
- $\Pr[M_{n+1} = l | M_n = l] = 1 - 2^{-l}$,
- $\Pr[M_{n+1} = l + 1 | M_n = l] = 2^{-l}$,

[2] Note, that one can initially set $M = 0$. However this subtle change causes the butterfly effect by complicating the proofs. Nevertheless, one can still choose the version with $M = 0$ and subtract 1 from the estimate.

for any $l \in \mathbb{N}$ *and* $n \in \mathbb{N}_0$.

Note that Definition 4.3.1 can be derived directly from a run of Algorithm 11. From now on, let $\Pr[M_n = l] =: p_{n,l}$. Directly from Definition 4.3.1 we get the following recursion:

$$p_{n+1,l} = (1 - 2^{-l})p_{n,l} + 2^{-l+1}p_{n,l-1} \tag{4.3}$$

for $l \in \mathbb{N}$ and $n \in \mathbb{N}_0$ with starting and boundary conditions $p_{0,1} = 1$, $p_{0,l} = 0$ for $l \geqslant 2$ and $p_{n,0} = 0$ for $n \in \mathbb{N}_0$. Let us signal that Eq. (4.3) is a core of proof of majority of lemmas about Morris Counter presented further.

Accuracy versus Differential Privacy

An accuracy of Morris Counter has been thoroughly analysed in various classical papers. First detailed analysis was proposed by Philippe Flajolet in [44]. Here we present the essence of theorems presented in [44], which will be useful later on:

Fact 4.3.3. *Let* M_n *denote Morris Counter after* n *successive incrementation requests. Then this random variable has an expected value* $\mathbb{E}[M_n] = \lg(n) - 0.273225\ldots$ *and variance* $\mathbb{V}\mathrm{ar}[M_n] = 0.763177\ldots$.

Realize that Fact 4.3.3 guarantees good accuracy[3] of Morris Counter and also high concentration of M_n around its average — a characteristic desirable in order to satisfy differential privacy definition. Fact 4.3.3 justifies the definition of moving intervals:

$$I_n = [\max(1, \lceil \lg(n) \rceil - 4) : \min(n + 1, \lceil \lg(n) \rceil + 4)] ,$$

which will emerge as a crucial point of our further considerations of this Markov process in terms of differential privacy in Section 4.5. Namely, we will show, that $M_n \in I_n$ with reasonable probability.

A lion's share of applications of Morris Counter is based on counting a number of occurrences, that is the number of incrementation requests. In order to estimate this value, we may use (4.3) and simply obtain $\mathbb{E}\big[2^{M_{n+1}}\big] = \mathbb{E}\big[2^{M_n}\big]+1$, so together with the assumption $M_0 = 1$ we obtain

$$\mathbb{E}\big[2^{M_n}\big] = n + 2 . \tag{4.4}$$

Hence $\Psi(M_n) = 2^{M_n} - 2$ is an unbiased estimator[4] of number of increments n. Remark that n can be saved in $\lceil \lg(n) \rceil$ bits. On the other hand, Fact 4.3.3 shows that on average, $\lg(\lg(n)) + \mathrm{O}(1)$ bits are required to save M_n. This is the most crucial advantage of Morris Counter. Moreover, analogically to (4.4) one may obtain

$$\mathbb{V}\mathrm{ar}\big[2^{M_n} - 2\big] = \frac{n(n+1)}{2} . \tag{4.5}$$

Formulas (4.4) and (4.5) will be used in the analysis of a data aggregation example in Section 4.6.

[3] See Section 1.3.4 for the definition of accuracy.

[4] An estimator is unbiased, if its expected value is exactly the estimated quantity.

4.4 MaxGeo Counter

We begin with a short description of MaxGeo Counter. Algorithm 12 shows an adaptation of its pseudo-code.

ALGORITHM 12: MaxGeo Counter Mechanism

```
   procedure Init()
1  |  C = []
2  |  M = 0
   procedure Aggregate()                    // on incrementation request
1  |  generate r ~ Geo_→(1/2)
2  |  append r to C
3  |  M = max(C)
   procedure Get()
1  |  return M
```

Speaking informally, we initialize an empty array C and set the MaxGeo Counter M to 0 (`Init`). Then, for each incrementation request, the server execute `Aggregate` to generate a random variable according to $\mathrm{Geo}_{\rightarrow}\left(\frac{1}{2}\right)$ distribution and append it to C. The final result is the maximum taken over all these generated random variables.

In differential privacy notion, the curator can publish the result via `Get` method.

Note that the variable M is in fact redundant, however it was left for clarity. When we erase the latter line both in `Init` and `Aggregate` and substitute the line of `Get` method by **return** $\max(C)$ in Algorithm 12, then we obtain a simpler version of pseudo-code.

Fact 4.4.1. *MaxGeo counter can be represented by the general counter from Definition 1.3.1 for $M_0 = 0$, $X(M_n) \sim \mathrm{Geo}_{\rightarrow}\left(\frac{1}{2}\right)$ and $f(x, y) = \max(x, y)$.*

The above fact shows that Algorithm 12 can be easily rewritten to save only the present maximum and freshly drawn number r instead of a (potentially big) array C and number r.

The expectation and variance of a maximum of n i.i.d. geometric variables have already been analysed in the literature. For instance, Szpankowski and Rego [99] provided exact formulas for expected value and variance of such the variables. However they are impractical for application fo bigger n. Hence they also provided asymptotics (here, for maximum of n independent $\mathrm{Geo}_{\rightarrow}\left(\frac{1}{2}\right)$ distributions): $\mathbb{E}[M_n] = \lg(n) + \mathrm{O}(1)$ and $\mathbb{V}\mathrm{ar}[M_n] \approx \frac{\pi^2}{6\ln(2)^2} + \frac{1}{12} = 3.507048\ldots$[5] and thus, similarly to Morris Counter, on average only $\lg(\lg(n)) + \mathrm{O}(1)$ are required to save MaxGeo Counter after n incrementation requests.

However, MaxGeo Counter was utilized as the aggregating algorithm by P. Flajolet and G.N. Martin in [46] for the first time. They calculated that

[5] A difference between the approximation of $\mathbb{V}\mathrm{ar}[M_n]$ and its exact value is a function with very small amplitude. For details, see [99].

$\mathbb{E}\left[\frac{2^{M_n}}{\varphi}\right] \approx n$,[6] where the magic "Flajolet—Martin constant" (the name according to [103]) is defined as follows:

$$\varphi = \frac{e^{\gamma}}{\sqrt{2}} \cdot \frac{2}{3} \prod_{n=1}^{\infty} \left(\frac{(4n+1)(4n+2)}{4n(4n+3)} \right)^{\epsilon_n} , \tag{4.6}$$

where $\gamma = 0,57721\ldots$ is Euler—Mascheroni constant (see Section 1.2.9) and ϵ_n is $\{-1,1\}$-Morse—Thue sequence (if $\nu(n)$ is the number of occurrences of digit '1' in BIN(n), then $\epsilon_n = (-1)^{\nu(n)}$).

4.4.1 Probabilistic Counting with Stochastic Averaging

Here we recall shortly more general Probabilistic Counting with Stochastic Averaging algorithm from [46].

Let m be of form 2^k, for some $k \in \mathbb{N}$. Assume that there are m, initially empty lots. For each incrementation request we connect it with one of the groups uniformly at random. Finally we perform Algorithm 12 separately and independently for each lot, obtaining MaxGeo Counters $M[1], M[2], \ldots, M[m]$. By $\varsigma_n(m)$ we denote a sum of these m MaxGeo Counters after total number of incrementation requests n. Let us introduce introduce an estimator:

$$\Xi_n(m) := \left\lfloor \frac{m}{\varphi} 2^{\frac{\varsigma_n(m)}{m}} \right\rfloor .$$

Then (according to [46]), for any $m = 2^k$, $k \in \mathbb{N}$,

$$\mathbb{E}[\Xi_n(m)] = n \left(1 + \frac{0.31}{m} + \psi_1(m,n) + o(1) \right), \text{ with } |\psi_1(m,n)| \leqslant 10^{-5} \tag{4.7}$$

and

$$\mathbb{V}\mathrm{ar}[\Xi_n(m)] = n^2 \left(\frac{0.61}{m} + \psi_2(m,n) + o(1) \right), \text{ with } |\psi_2(m,n)| \leqslant 10^{-5} . \tag{4.8}$$

Note that averaging reduces the variance of the probabilistic counter. Remark that "Stochastic Averaging" in PCSA algorithm refers to the random choice of the numbers of entities in each group and it slightly differs from the standard averaging solution via Monte Carlo method with groups of equal size.

4.4.2 HyperLogLog

The maximum of geometric variables is used as a primitive in well known HyperLogLog algorithm (see [45]), therefore its privacy properties are important both from theoretical and practical point of view. See that essentially in HyperLogLog we have k independent MaxGeo counters $M[1], \ldots, M[k]$ and for

[6] Here the approximation is given with respect to some periodic multiplier of amplitude less than 10^{-5} and mean value 1.

each incrementation request we choose one of the counters uniformly at random. Let us denote the chosen counter by $M[j]$. Then we generate random variable with $X \sim \mathrm{Geo}_{\rightarrow}\left(\frac{1}{2}\right)$ distribution and update $M[j] := \max(M[j], X)$. The final estimation is

$$\mathrm{HyperLogLog} := \alpha_k k^2 \left(\sum_{j=1}^{k} 2^{-M[j]} \right)^{-1},$$

where α_k is a constant dependent only on k (see details in [45]). It worth to note that HyperLogLog related algorithms (mentioned in Section 4.1) are the best known procedures designated for cardinality estimation and they are close to optimum (according to [62]).

According to [45], for $m = 2^k$, where $k \geqslant 4$,

$$\mathbb{E}[\mathrm{HyperLogLog_n}] = n(1 + \psi_3(n) + o(1)), \text{ with } |\psi_3(n)| < 5 \cdot 10^{-5}$$

and

$$\mathbb{V}\mathrm{ar}[\mathrm{HyperLogLog_n}] = n^2 \left(\frac{\beta_m}{\sqrt{m}} + \psi_4(n) + o(1) \right)^2, \text{ with } |\psi_4(n)| < 5 \cdot 10^{-4},$$

where $\beta_m \overset{m\to\infty}{\longrightarrow} \sqrt{2\lg(2) - 1} = 1.03896\ldots$ and $\beta_m \leqslant 1.106$ for $m \geqslant 16$.

4.5 Probabilistic Counters Privacy Properties

4.5.1 Morris Counter Privacy

In this subsection we investigate Morris Counter in terms of (ε, δ)-DP in order to obtain the following

Theorem 4.5.1. *Let M denote Morris Counter and assume $|n-m| \leqslant 1$. Then*

$$\Pr[M_n = l] \leqslant \left(1 - \frac{16}{n}\right)^{-1} \cdot \Pr[M_m = l] + \delta,$$

where $\delta < 0.00033$, so M is $(L(n), 0.00033)$-DP with

$$L(n) = -\ln\left(1 - \frac{16}{n}\right) = \frac{16}{n} + \frac{128}{n^2} + \mathrm{O}\left(n^{-3}\right) \leqslant \frac{16}{n-8}.$$

To do so, we take the following steps. First we consider a concentration of Morris Counter in the vicinity of its mean value. More precisely, we show that Morris Counter after n incrementation requests takes values in relatively small intervals I_n with probability at least $1 - \delta$ (then M_n satisfies the condition (4.2) for $S(n) = I_n$), where δ is some small constant arose from the proof. Note that I_n may be interpreted as confidence intervals at level $1 - \delta$ (see e.g. [86]). Afterwards, we provide some lemmas which let us establish a bound for $\varepsilon(n)$ parameter in formula (4.1) in interval I_n for $n > 2^7$. Finally we obtain

analogous results for smaller number of incrementation requests n numerically (the argument for $n > 2^7$ from the proof does not extrapolate for smaller n, although the claim remains true).

The following theorem, provided by Flajolet, will be useful in our reasoning:

Theorem 4.5.2 (Proposition 1 from [44])**.** *The probability $p_{n,l}$ that the Morris Counter has value l after n incrementation requests is*

$$p_{n,l} = \sum_{j=0}^{l-1} (-1)^j \, 2^{-j(j-1)/2} \left(1 - 2^{-(l-j)}\right)^n \prod_{i=1}^{j} \left(1 - 2^{-i}\right)^{-1} \prod_{i=1}^{l-1-j} \left(1 - 2^{-i}\right)^{-1} \ .$$

Notice that Theorem 4.5.2 presents an explicit formula for $\Pr[M_n = l]$, which (as we may experience in Appendix E) is not convenient to analyze. However it is simple enough to find the values numerically (also note that recursive Definition 4.3.1 provides those probabilities as well, but this approach is not efficient in terms of memory and time for big number of incrementation requests n). We are going to use Theorem 4.5.2 in technical proofs of lemmas in Appendix E.

Let us commence with a reminder. First of all, M_n is ranged in $\mathbb{N}_0$ and moreover $I_n \subset [\lceil \lg(n) \rceil - 4 : \lceil \lg(n) \rceil + 4]$. We provide few facts about the concentration of the distribution of the random variable M_n, or more precisely the probability that M_n will be outside the interval I_n:

Lemma 12. *Let M_n be the state of the Morris Counter after n incrementation requests. Then*

$$\delta_1 := \Pr[M_n \leqslant \lceil \lg(n) \rceil - 5] \leqslant 0.000006515315 \ldots \ .$$

Lemma 13. *Let M_n be the state of the Morris Counter after n incrementation requests. Then*

$$\delta_2 := \Pr[M_n \geqslant \lceil \lg(n) \rceil + 5] \leqslant 0.000325521 \ldots \ .$$

The proofs of both lemmas 12 and 13 are presented in Appendix E.

Theorem 4.5.3. *The state of the Morris Counter after n incrementation requests is **not** in a set I_n with probability $\delta < 0.00033$.*

Proof. Realize that $\Pr[M_n \in [n+1]] = 1$. This observation, together with lemmas 12 and 13 bears a conclusion that

$$\delta := \Pr[M_n \notin I_n] = \delta_1 + \delta_2 < 0.00033 \ .$$

□

In the second part of the investigation, we try to establish $\varepsilon(n)$ parameter of DP property of M_n. In fact, it remains to examine the property (4.1) in the interval I_n, since Theorem 4.5.3 entails (4.2) for $S_n = I_n$. Therefore, we are

interested in finding the upper bound for *maximal privacy loss* for any $n \in \mathbb{N}$ and $k \in I_n$:

$$\varepsilon(n) = \max\left(\left|\ln\left(\frac{p_{n\pm1,k}}{p_{n,k}}\right)\right| : k \in I_n\right) . \tag{4.9}$$

Actually, we may consider $'+'$ sign instead of $\pm$ in (4.9), because $|\ln(x)| = \left|\ln\left(\frac{1}{x}\right)\right|$. However, when $I_n \neq I_{n\pm1}$, we have to behave carefully, so in particular, an additional check of privacy loss with $'-'$ sign is needed when n is of a form $2^l + 1$ for some $l \in \mathbb{N}$.

Claim 1. *For $k \geqslant 7$, we have $p_{2^k+1,k+4} \leqslant 2^7 p_{2^k+1,k+5}$.*

The above claim is result of a simple application of Lemma 26 from Appendix E.

Claim 2. *If for any given n, there exists an ascending and positive sequence $(\alpha_i)_{i=1}^n$ such that $(\forall\, i \in [n])\ p_{n,i} = 2^i \alpha_i p_{n,i+1}$, then there also exists an ascending and positive sequence $(\alpha'_i)_{i=1}^{n+1}$ such that*

$$(\forall\, i \in [n+1])\ (p_{n+1,i} = 2^i \alpha'_i p_{n+1,i+1}) \wedge (\forall\, i \in [n])\ (\alpha'_i < \alpha_i) .$$

This claim emerges from lemmas 27 and 28 from Appendix E. We are going to use Claim 2 in order to guarantee appropriate starting conditions for the next Lemma 14 (i.e. we would like to provide minimal $l \in \mathbb{N}$ such that $n = 2^{l-1}$ and $p_{n+1,l+c-1} \leqslant 2^{c+3} p_{n+1,l+c}$ for every c in an interval $[-l+1 : 4]$). Moreover, let us mention that in order to apply Lemma 14 we are going to use Claim 1 from Appendix E as well. Claim 1 assumes that $n \geqslant 2^7 + 1$, therefore we would like to gather some information about the distribution of M_{2^7+1}, before we are going to apply our milestone results. Namely, we are interested in the behaviour of $\theta_i = \dfrac{p_{129,i}}{p_{129,i+1}}$ for $i \leqslant 11$ (presented in the Table 4.1). In Table 4.1 we briefly

i	θ_i	2^{i-4}	$2^{4-i}\theta_i$
1	$9.6205\ldots \cdot 10^{-24}$	0.125	$7.6964\ldots \cdot 10^{-23}$
2	$1.73351\ldots \cdot 10^{-9}$	0.25	$6.93402\ldots \cdot 10^{-9}$
3	$0.000119359\ldots$	0.5	$0.000238718\ldots$
4	$0.0140238\ldots$	1	$0.0140238\ldots$
5	$0.158163\ldots$	2	$0.0790814\ldots$
6	$0.771817\ldots$	4	$0.192954\ldots$
7	$2.67702\ldots$	8	$0.334628\ldots$
8	$7.83367\ldots$	16	$0.489604\ldots$
9	$20.8095\ldots$	32	$0.650297\ldots$
10	$52.0472\ldots$	64	$0.813238\ldots$
11	$125.065\ldots$	128	$0.977073\ldots$

Table 4.1: Ratios of probabilities of adjacent atoms of distribution of M_{2^7+1} variable, compared with the exponential function of the base 2.

recognize at least exponential trend of proportions θ_i, so we are able to use

Claim 2 for $n \geqslant 2^7 + 1$. At the first glance, the choice of n seems arbitrary, but it occurs that a distribution of M_{2^6+1} does not preserve necessary properties of privacy loss (note that nevertheless M_{2^6+1} still is able to satisfy the property of $(\varepsilon(n), \delta)$-DP with parameters given in Theorem 4.5.1).

Lemma 14. *Let $l \in \mathbb{N}$ and $n = 2^{l-1}$. If $p_{n+1,l+c-1} \leqslant 2^{c+3} p_{n+1,l+c}$ for every c in an interval $[-l+1:4]$, then*

$$(\forall\, N \geqslant n+1)(\forall\, c \in [-l+1:4])\; p_{N,l+c-1} < 2^{c+3} p_{N,l+c}\ .$$

Proof. Realize that for $c = -l+1$, the required inequality is a straightforward conclusion from Definition 4.3.1. Therefore we can safely consider $c \in [-l+2:4]$ and any $N \geqslant n+1$ with an assumption that $p_{N,l+d-1} < 2^{d+3} p_{N,l+d}$ for $d \in \{c-1, c\}$. Then

$$\begin{aligned} p_{N+1,l+c-1} &\overset{(4.3)}{=} p_{N,l+c-1}(1 - 2^{-l-c+1}) + p_{N,l+c-2} 2^{-l-c+2} \\ &\leqslant 2^{3+c} p_{N,l+c}(1 - 2^{-l-c+1}) + 2^{3+(c-1)} p_{N,l+c-1} 2^{-l-c+2} \\ &< 2^{3+c}(p_{N,l+c}(1 - 2^{-l+c}) + 2^{-l-c+1} p_{N,l+c-1}) = 2^{3+c} p_{N+1,l+c}\ . \end{aligned}$$

The thesis is the result of an appropriate application of inductions. □

Claims 1 and 2, together with Table 4.1 enable us to apply Lemma 14 for $n = 2^k + 1$ for any $k \geqslant 7$.

Theorem 4.5.4. *Let $n > 2^7 = 128$ and $k \in I_n$. Then*

$$1 - \frac{16}{n} \leqslant \frac{p_{n\pm1,k}}{p_{n,k}} \leqslant 1 + \frac{16}{n}\ .$$

Proof. According to the previous discussion about the formula (4.9), we examine:

$$\frac{p_{n+1,k}}{p_{n,k}} \overset{(4.3)}{=} \frac{p_{n,k}(1 - 2^{-k}) + 2^{-k+1} p_{n,k-1}}{p_{n,k}} = 1 + 2^{-k}\left(-1 + 2\frac{p_{n,k-1}}{p_{n,k}}\right)\ .$$

Let us denote $l = \lceil \lg(n) \rceil$ and $c = k - l \in [-4:4]$. Then Lemma 14 bears $p_{n,k-1} \leqslant 2^{c+3} p_{n,k}$, so

$$\frac{p_{n+1,k}}{p_{n,k}} \leqslant 1 + 2^{-l-c}(-1 + 2^{c+4}) < 1 + 2^{-l+4} = 1 + \frac{16}{2^{\lceil \lg(n) \rceil}} < 1 + \frac{16}{n}\ .$$

Realize that if $n = 2^{l-1} + 1$ for some $l \in \mathbb{N}$, then a little adjustment is necessary. Indeed, let now $c - 1 = k - l \in [-4:4]$, and once again, Lemma 14 provides $p_{n-1,k-1} < 2^{c+2} p_{n-1,k}$. However, it still holds that:

$$\frac{p_{n,k}}{p_{n-1,k}} = 1 + 2^{-k}\left(-1 + 2\frac{p_{n-1,k-1}}{p_{n-1,k}}\right) \leqslant 1 + 2^{-l-c+1}\left(-1 + 2^{c+3}\right) < 1 + \frac{16}{n}\ .$$

On the other hand, we obviously have inequalities $p_{n+1,k} > \left(1 - 2^{-l-c}\right) p_{n,k}$ and $p_{n,k} > \left(1 - 2^{-l-c}\right) p_{n-1,k}$ for any $c \in [-4:4]$, so in both cases, $1 - 2^{-l-c}$ exceed $1 - \frac{16}{n}$. □

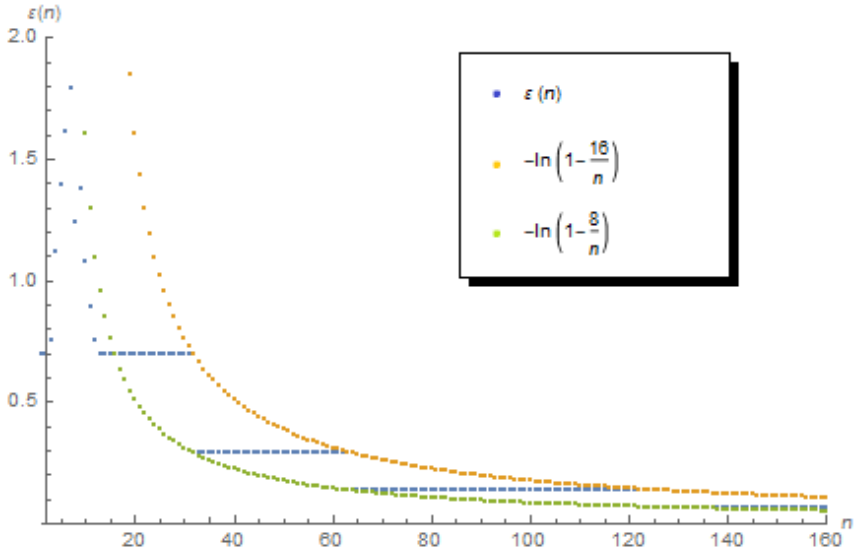

Figure 4.1: Exact values of $\varepsilon(n)$ parameter for $n \leqslant 160$ compared with plots of sequences $-\ln\left(1-\frac{16}{n}\right)$ and $-\ln\left(1-\frac{8}{n}\right)$.

Remark that Theorem 4.5.4 only provides $\varepsilon(n) \leqslant -\ln\left(1-\frac{16}{n}\right)$ for $n > 128$ (compare with (4.9)). However, in Figure 4.1 we present the first 160 exact values of $\varepsilon(n)$, defined by Eq. (4.9) compared with sequences $\left(-\ln\left(1-\frac{16}{n}\right)\right)_{n=1}^{160}$ and $\left(-\ln\left(1-\frac{8}{n}\right)\right)_{n=1}^{160}$. We may briefly see that inequality $\varepsilon(n) \leqslant -\ln\left(1-\frac{16}{n}\right)$ is still satisfied for smaller number of requests n as well. Moreover, in almost all of the cases $\varepsilon(n) > -\ln\left(1-\frac{8}{n}\right)$ (especially for $n > 16$). We can also observe that $\varepsilon(n) \approx 2^{4-\lceil \lg(n) \rceil}$ in this interval. Note that $\lceil \lg(n) \rceil \leqslant 4$ for $n \leqslant 16$, so $\lceil \lg(n) \rceil - 4 < 1$, but M is always positive. This can be seen as a reason of chaotic behaviour of the process for $n \leqslant 16$. Nevertheless, Figure 4.1 affirms the quality of $\varepsilon(n)$ parameter established in Theorem 4.5.1. Moreover, we present the following

Observation 2. *The constant* 16 *in Theorem 4.5.1 cannot be improved. See that*

$$\frac{p_{33,1}}{p_{32,1}} = \frac{1}{2} = 1 - \frac{16}{32} \ .$$

Having all the technical lemmas, we are finally ready to prove Theorem 4.5.1.

Proof. Suppose that $S(n) = I_n$ in Fact 4.3.1. Then from theorems 4.5.3 and 4.5.4 we can easily see that $\Pr[M_n \notin S(n)] < 0.00033$ and

$$(\forall\, m \in \{n-1, n+1\})\ (\forall\, l \in S(n))\ \Pr[M_n = l] \leqslant \left(1 - \frac{16}{n}\right)^{-1} \cdot \Pr[M_m = l] \ ,$$

hence from Fact 4.3.1 we obtain the main result. □

4.5.2 MaxGeo Counter Privacy

In this subsection we present a theorem which shows the privacy guarantees of MaxGeo Counter. Assume that we have n incrementation requests. In case of MaxGeo Counter, it means that we generate random variables $X_1, \ldots, X_n$, where $X_i \sim \mathrm{Geo}_{\rightarrow}\left(\frac{1}{2}\right)$ are pairwise independent. Ultimately the result of the counter is maximum over all X_i's, namely $X = \max(X_1, \ldots, X_n)$. We present the following

Theorem 4.5.5. *Let M denote the MaxGeo counter and n denote the number of incrementation requests. Consider m such that $|n - m| \leqslant 1$. Fix $\varepsilon > 0$ and $\delta \in (0, 1)$ and let*

$$l_\varepsilon = \left\lceil \log\left(\frac{e^\varepsilon}{e^{-\varepsilon} - 1}\right)\right\rceil \ .$$

If

$$n \geqslant \frac{\ln(\delta)}{\ln\left(1 - 2^{-l_\varepsilon}\right)} \left(\approx -\frac{\ln(\delta)}{\varepsilon}\right) \ , \tag{4.10}$$

then

$$\Pr[M_n \in S] \leqslant e^\varepsilon \cdot \Pr[M_m \in S] + \delta,$$

so M is (ε, δ)-DP.

Proof. After n incrementation requests for MaxGeo Counter M, the result of the mechanism is $X = \max(X_1, \ldots, X_n)$, where $X_i \sim \mathrm{Geo}_{\rightarrow}\left(\frac{1}{2}\right)$ are pairwise independent. First we observe that if $n = m$ then the counter trivially satisfies differential privacy, as the probability distribution of X does not change at all. From now on we assume that $|n - m| = 1$. See that

$$\Pr[X \leqslant l] = \prod_{i=1}^{n} \Pr[X_i \leqslant l] = (\Pr[X_1 \leqslant l])^n = \left(1 - \frac{1}{2^l}\right)^n = \left(\frac{2^l - 1}{2^l}\right)^n \ .$$

Furthermore

$$\begin{aligned}\Pr[\max(X_1, \ldots, X_n) = l] = \Pr[X = l] &= \Pr[X \leqslant l] - \Pr[X \leqslant (l-1)] \\ &= \left(\frac{2^l - 1}{2^l}\right)^n - \left(\frac{2^{l-1} - 1}{2^{l-1}}\right)^n = \frac{\left(2^l - 1\right)^n - \left(2^l - 2\right)^n}{2^{l \cdot n}} \ .\end{aligned}$$

Now we need to calculate the following expression

$$\frac{\Pr[\max(X_1, \ldots X_n) = l]}{\Pr[\max(X_1, \ldots X_n, X_{n+1}) = l]} = \frac{\frac{\left(2^l - 1\right)^n - \left(2^l - 2\right)^n}{2^{l \cdot n}}}{\frac{\left(2^l - 1\right)^{n+1} - \left(2^l - 2\right)^{n+1}}{2^{l \cdot (n+1)}}} = \ldots$$

$$\begin{aligned}
\ldots &= \frac{\left(\left(2^l-1\right)^n-(2^l-2)^n\right)\cdot 2^l}{\left(2^l-1\right)^{n+1}-(2^l-2)^{n+1}} \\
&= \frac{2^l}{2^l-1}\cdot\frac{\left(\left(2^l-1\right)^n-(2^l-2)^n\right)}{\left(\left(2^l-1\right)^n-\frac{(2^l-2)^{n+1}}{2^l-1}\right)} \\
&\leqslant \frac{2^l}{2^l-1}\cdot\frac{\left(\left(2^l-1\right)^n-(2^l-2)^n\right)}{\left(\left(2^l-1\right)^n-\frac{(2^l-2)^{n+1}}{2^l-2}\right)} \\
&= \frac{2^l}{2^l-1} = 1+\frac{1}{2^l-1}\ .
\end{aligned}$$

For fixed ε value we need to satisfy the following inequality

$$\left|\ln\left(\frac{\Pr[\max(X_1,...,X_n)=l]}{\Pr[\max(X_1,...,X_n,X_{n+1})=l]}\right)\right| \leqslant \varepsilon\ ,$$

which gives

$$\ln\left(1+\frac{1}{2^l-1}\right) \leqslant \varepsilon\ . \tag{4.11}$$

From (4.11) we can see that the greater l is, the smaller ε can be. Moreover, inequality (4.11) is true for $l \geqslant l_\varepsilon$. Therefore, we have to assure that $\Pr[X \leqslant l_\varepsilon] \leqslant \delta$. See that

$$\Pr[X \leqslant l_\varepsilon] = \left(1-2^{-l_\varepsilon}\right)^n\ .$$

It is easy to see that the above expression decreases with respect to n. Therefore

$$\left(1-2^{-l_\varepsilon}\right)^n \leqslant \delta \iff n \geqslant \frac{\ln(\delta)}{\ln(1-2^{-l_\varepsilon})} \approx -\frac{\ln(\delta)}{\varepsilon},$$

where the approximation is a result of substitution of l_ε without ceiling. □

Ultimately it means that for fixed privacy parameters (ε,δ) we can calculate the minimum number of incrementation requests that is necessary to satisfy given privacy parameters. This can be done by artificially adding them before actually collecting data. Of course it has to taken into account that the initially added value should be subtracted from the final estimation of the appropriate cardinality before a publication and this change can impact on the precision of the estimation. See that if we can perform such a preprocessing, then for every (ε,δ) we can easily know how many artificial counts have to be added.

Note that from differential privacy perspective, HyperLogLog is an arbitrary postprocessing performed on k MaxGeo counters. Moreover, as each response goes to one counter only, they are independent from each other, which means that we can use parallel composition theorem (see [40]). This gives us the following

Observation 3. *Assume we have k MaxGeo counters $M[1],\ldots,M[k]$, which are used in HyperLogLog algorithm. If j-th MaxGeo counter is (ε_j,δ_j)-DP then HyperLogLog is $(\max_i \varepsilon_i, \max_i \delta_i)$-DP.*

4.5.3 Comparison of Morris and MaxGeo Counters' Privacy

In this subsection we compare privacy and storage properties of data aggregation algorithms based on one of the investigated counters or standard Laplace method.

We start with auxiliary remarks for the privacy of MaxGeo Counter. For instance, see that for fixed δ and n we may obtain from Theorem 4.5.5 that

$$\varepsilon(n) \geqslant \left(2^{\left\lfloor -\lg\left(1-\delta^{\frac{1}{n}}\right)\right\rfloor} - 1\right)^{-1} =: \varepsilon_0(n) \ . \tag{4.12}$$

We want to optimize $\varepsilon(n)$, so we are going to consider $\varepsilon_0(n)$ defined as the right hand side of (4.12). In order to limit let us consider the following function of $x \in \mathbb{R}_+$:

$$\psi(x,\delta) := \left(\left(1-\delta^{\frac{1}{x}}\right)^{-1} - 1\right)^{-1} = -\frac{\ln(\delta)}{x} + \frac{\ln(\delta)^2}{2x^2} - \frac{\ln(\delta)^3}{6x^3} + \mathrm{O}\left(x^{-4}\right) \ , \tag{4.13}$$

as $x \to \infty$, so $\varepsilon(n) \geqslant \psi(n,\delta) = -\frac{\ln(\delta)}{n} + \mathrm{O}\left(n^{-2}\right)$, as $n \to \infty$. Since ψ is decreasing with respect to x, we are going to consider when $\varepsilon_0(n)$ changes. More precisely, consider a minimal k such that $\varepsilon_0(n) < \psi(n-k,\delta) \leqslant \varepsilon_0(n-k)$, which occurs to be the neat upper bound for $\varepsilon(n)$. However, since $\varepsilon_0(n)$ is the non-ascending step function, we realize that

$$\varepsilon_0(n-k) \geqslant \left(2^{\left\lfloor -\log\left(1-\delta^{\frac{1}{n}}\right)\right\rfloor - 1} - 1\right)^{-1} = -\frac{2\ln(\delta)}{n} + \frac{3\ln(\delta)^2}{n^2} - \frac{13\ln(\delta)^2}{3n^3} + O(n^{-4}) \, ,$$

as $n \to \infty$. If we denote $\phi(n,\delta) := \left(2^{\left\lfloor -\log\left(1-\delta^{\frac{1}{n}}\right)\right\rfloor - 1} - 1\right)^{-1}$, then we can sum up our recent considerations shortly by: $\psi(n,\delta) \leqslant \varepsilon(n) < \phi(n,\delta)$. Thence, in the case when we fix parameter $\delta = 0.00033$, we obtain

$$\frac{8.0164\ldots}{n} + \frac{32.13147\ldots}{n^2} + \mathrm{O}\left(n^{-3}\right) \leqslant \varepsilon(n) \leqslant \frac{16.0328\ldots}{n} + \frac{192.789\ldots}{n^2} + \mathrm{O}\left(n^{-3}\right) ,$$

as $n \to \infty$. On the other hand, from Theorem 4.5.1, we know that when $\delta = 0.00033$, then for Morris Counter (with $\varepsilon(n)$ defined by (4.9)) the quite similar relation holds:

$$\varepsilon(n) \leqslant -\ln\left(1 - \frac{16}{n}\right) = \frac{16}{n} + \frac{128}{n^2} + \mathrm{O}\left(n^{-3}\right) \ ,$$

as $n \to \infty$. This shows that both Morris Counter and MaxGeo Counter behave quite similarly under comparable conditions and Figure 4.2 confirms this observation. Indeed, in Figure 4.2 we may see that the difference between the values of $\varepsilon(n)$ parameters for both counters shrinks as n gets bigger.

Realize that previous conclusions remain true if $\delta(n)$ is not constant. This short observation enable us to obtain a more general result:

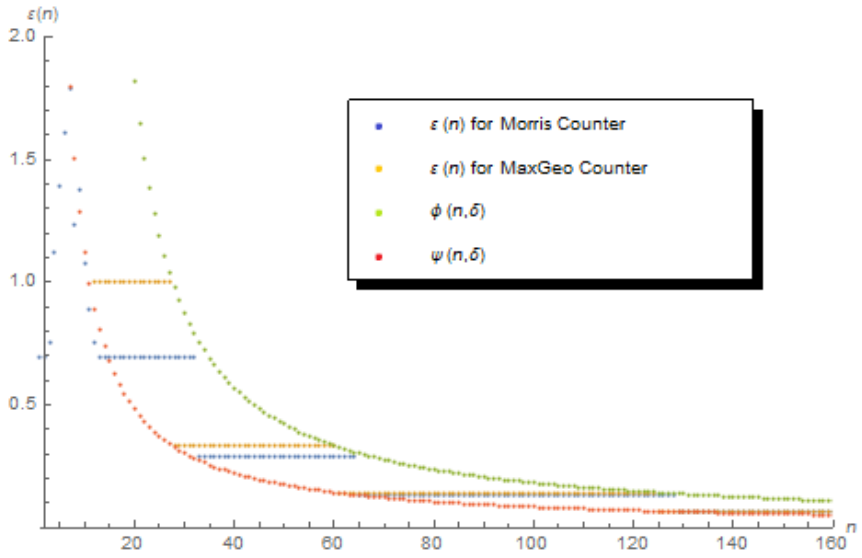

Figure 4.2: Values of $\varepsilon(n)$ parameters for Morris and MaxGeo Counters compared with boundaries for $\varepsilon(n)$ for MaxGeo Counter: the lower one — $\psi(n, \delta)$ and the upper one — $\phi(n, \delta)$ ($n \leqslant 160$ and $\delta = 0.00033$).

Example 11. *Let $\delta(n) = n^{-c}$ for some constant $c > 0$. Then*

$$\varepsilon(n) \leqslant \phi(n, \delta(n)) = \frac{2c\ln(n)}{n} + \frac{3c^2\ln(n)^2}{n^2} + \mathrm{O}\left(\frac{c^3\ln(n)^3}{n^3}\right)$$

as $n \to \infty$, so MaxGeo Counter is $(\phi(n, \delta(n)), \delta(n))$-DP for any $n \in \mathbb{N}$. Both sequences of parameters tend to 0, which may be used as an advantage when an expected number of incrementation requests D is very large. However, we emphasize that this requires $\delta(n)$ to be negligible.

4.6 Private Survey via Probabilistic Counters

In this section we present an example scenario for data aggregation using probabilistic counters. We assume that there is a *server* (alternatively we call it *aggregator*) and a collection of *nodes* (e.g. mobile phone users) and we want to perform a boolean survey with a sensitive question. Namely, each user sends '0' if his/her answer is *no* and '1' if the answer is *yes*. We assume that the connections between users and the server are perfectly secure and the data can safely get to the trusted server. This can be performed using standard cryptography solutions. The goal of the server is to publish the sum of all '1' responses in a privacy-preserving way. Such goal could obviously be achieved by simply collecting all the data and adding an, appropriately calibrated, Laplace noise

(see [40]), but we aim to show that probabilistic counters have inherently sufficient randomness to be differentially private, without any auxiliary randomizing mechanism.

We can present the general scenario in the following way:

1. each user sends his/her bit of data to the server using secure channels,
2. server plugs the data points sequentially into the counter,
3. if the data point is '1', the counter receives *incrementation request*, otherwise, the data is ignored,
4. each incrementation request is being processed by the counter and may lead (depending on randomness) to an increase of the value of the counter,
5. when all data is processed, the value of the counter is *released* to the public.

Note that we assume that the adversary has access **only** to the released value. See also that we released just the counter value itself, which is not actually an estimation of '1' responses. Such estimation is a function of released value, which is different for Morris or MaxGeo Counter and there also can be various ways to estimate the actual number using counter value. However, this does not really matter for our case, as differential privacy is, conveniently, fully resilient to post-processing (see [40]). The graphical depiction of our considered scenario is presented in Figure 4.3.

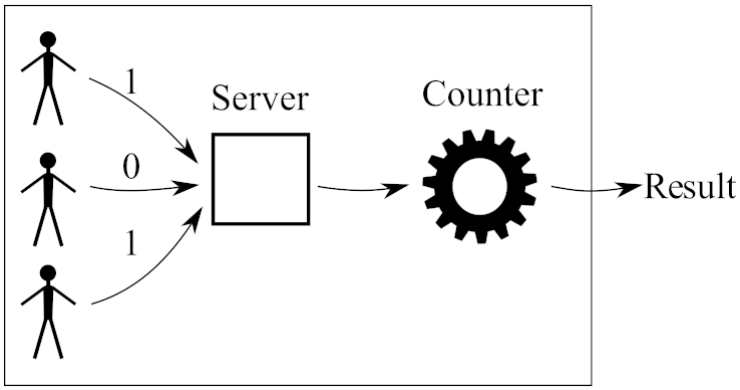

Figure 4.3: Scenario for data aggregation using probabilistic counters. We assume that the Adversary does not have any way to extract information from within the rectangle.

Adversary Our assumptions about the adversary are the same as in most differential privacy papers. Namely, he may collude with any subset of the participants (e.g. all except the single user whose privacy he wants to breach).

On the other hand, the aggregator is trusted. See that even though we have a distributed system in mind, this is, in fact, a central differential privacy scenario. We do not assume pan-privacy. It means that the internal state of the algorithm is **not** subject to the constraints of differential privacy. Obviously if the adversary would know the internal state of the counter at any time or could observe whether after receiving data from specific user the server has to perform computations to potentially increment the counter (implying a '1' response) or not, he would easily violate the privacy. We also do not assume privacy under continual observation, the survey is not published iteratively, but one time only, after it is finished. To sum it up, the adversary **cannot**

- extract or tamper with the internal state of the counter,
- extract any information from the server or channels between users and server.

The adversary **can**

- collude with any subset C of the participants (e.g. know their data or make them all send '0' to the server) in order to breach privacy of user not belonging to C,
- obtain the final result of the aggregation and perform any desired post-processing on it.

Note that, in light of our theorems 4.5.1 and 4.5.5 both Morris Counter and MaxGeo Counter preserve differential privacy in such scenario. Assume we have at least n users holding '1', therefore at least n incrementation requests. See that we can either know it based on domain knowledge (e.g. we expect that at least some fraction of users will send '1' based on similar surveys) or add artificial n counts to the counter initially. Obviously, in case of artificial counts it has to be taken into account when estimating the final sum. Using Morris Counter we obtain $(L(n), 0.00033)$-DP with

$$L(n) = -\ln\left(1 - \frac{16}{n}\right) \leqslant \frac{16}{n-8} \ .$$

We present the following

Example 12. *Assume we have at least $n = 200$ incrementation requests. From Theorem 4.5.1, we have $L(n) \leqslant \frac{16}{n-8} \leqslant 0.08334$. Therefore, using Morris Counter, the survey presented above is $(0.08334, 00033)$-DP.*

On the other hand, using MaxGeo Counter for a given ε and δ we get (ε, δ)-DP as long as $n \geqslant \frac{\ln(\delta)}{\ln(1-2^{-l_\varepsilon})}$, where $l_\varepsilon = \left\lceil \log\left(\frac{e^\varepsilon}{e^{-\varepsilon}-1}\right)\right\rceil$. Here we present an example.

Example 13. *Let $\varepsilon = 0.5$ and $\delta = \frac{1}{D^2}$, where D is the the number of all survey participants. After using Theorem 4.5.5 and straightforward calculations we have $n \geqslant 7\ln(D)$. Say we will have $e^{20} = 4.85165\ldots\cdot 10^8$ participants. Then if we have at least 140 incrementation requests, we satisfy $\left(0.5, \frac{1}{D^2}\right)$-DP.*

Method	Laplace noise $\mathcal{L}\left(\frac{n}{16}\right)$	Morris counter M_n	m MaxGeo counters $\left(M_n^{(i)}\right)_{i=1}^{m}$
(ε, δ)-DP	$\left(\frac{16}{n}, 0\right)$-DP	$(\frac{16}{n-8}, \delta)$-DP	$(\sim \frac{16.033}{n}, \delta)$-DP
Estimator $\Psi(n)$	$n + \mathcal{L}\left(\frac{n}{16}\right)$	$2^{M_n} - 2$	$\left\lfloor \frac{m}{\varphi} 2^{\frac{\varsigma_n(m)}{m}} \right\rfloor$
$\mathbb{V}\mathrm{ar}[\Psi(n)]$	$\frac{n^2}{128}$	$\frac{n^2+n}{2}$	$\sim \frac{0.78}{m} \cdot n^2$
Avg. memory	$\lg(n) + \mathrm{O}(1)$	$\lg(\lg(n)) + \mathrm{O}(1)$	$m \lg\left(\lg\left(\frac{n}{m}\right)\right) + \mathrm{O}(1)$

Table 4.2: A juxtaposition of data aggregation techniques. The standard one is based on Laplace method and the rest are based on probabilistic counters. Recall that $\delta = 0.00033$ and $\varphi = 0.77351\ldots$ ($\mathrm{O}()$ terms are provided for $n \to \infty$).

In Table 4.2 we present a summary of three mechanisms with comparable differentially private properties. The first one is based on Laplace method and the other two are based on probabilistic counters. The first one has the biggest memory use, however it is the most accurate. We may briefly see that probabilistic counters may be used for data aggregation in order to decrease the memory usage by a cost of small positive δ parameter of differential privacy and a slightly bigger variance, which equivalently worsen the accuracy. Notice that both counters performs very similar, however Morris Counter is better, since we have to bear in mind, that one can use a mean of independent estimators of several probabilistic counters in order to improve the accuracy in exchange for an increase of memory cost (proportionally to the number of used counters). This argument justifies the usability of such the mechanisms, when one is interested in very memory-efficient protocols, which are quite accurate and first of all – differentially private.

Example 14. *Imagine that 100 million people take part in a general health survey with* 100 *yes/no sensitive questions. For every question, we would like to provide an estimation of the number of people who answered yes, but we want to guarantee differential privacy property at reasonable level. Realize that if the number of yes answers is very small for some questions (e.g. when the question refers to a very rare disease), then the number of no answers may be counted instead. According to Table 4.2, if we use Laplace method, then we may need approximately* $100 \lg(10^8) = 2657.54\ldots$ *bits to store the counters. However, if we use Morris Counter instead, then about* $100 \lg(\lg(10^8)) = 473.20\ldots$ *bits are needed.*

One may complain about a heavy use of pseudo-random number generator that probabilistic counters make. However this problem may be resolved by generating the number of incrementation requests, which have to be forgotten until the next update of the counter by utilizing appropriate geometric distributions (compare with jumps in Section 3.2 or see e.g. [74] for similar approach

applied to reservoir sampling algorithm). This way the use of PRNG can be substantially reduced.

4.7 Previous and Related Work

In this chapter we take a unusual perspective on probabilistic counters. Namely, we focus on their inherent privacy guarantees in the sense of differential privacy. The idea of differential privacy has been introduced for the first time in [38], however its precise formulation in the widely used form appeared in [35]. There is a long list of papers concerning differential privacy, e.g. [36, 37, 39]. Most of these papers focus on a centralised model, namely a database with trusted party holding it. See that in our paper, despite the distributed setting, we have the same trust model. The curator is entitled to gather and see all participants' data in the clear and releases the computed data to wider (possibly untrusted) audience. Comprehensive information concerning differential privacy can be found in [40].

The idea of probabilistic counters, along with the well known Morris Counter was presented in the seminal paper [80]. Very detailed analysis by Flajolet can be found in [44]. MaxGeo Counter was first proposed and analysed in [99]. More detailed and precise analysis can be found in [42]. Most important application of MaxGeo Counter can be found in [45], where the authors propose the well-known HyperLogLog algorithm. Its practical applications are widely described in [58]. There are several widely used improvements of HyperLogLog algorithm: HyperLogLog+ [58], Streaming HyperLogLog with sketches based either on historic inverse probability [25] or martingal estimator [101] or empirically adjusted HyperBitBit (proposed by R.Sedgewick [94]). A main goal of these adjustments is a reduction of the memory requirements (see e.g. [109] or [102]). For instance, some of the above solutions are used in database' systems for queries' optimization or for document classification purpose. Moreover, MaxGeo counter was used in [88], for an adjustment of ANF tool, developed for data mining from extensive graphs, which enables to answer many different questions, based on some neighbourhood function defined on the graph.

Probabilistic counters were previously considered in terms of privacy preservation for set cardinality estimators. The authors of [33] show that in the scenario of using probabilistic counters for set cardinality estimation with the adversary being able to extract the intermediate values of counter, the privacy is not preserved. In this chapter we perform data aggregation instead of cardinality estimation, moreover we assume that the adversary is **not** able to extract any intermediate values from the counter. To the best of authors knowledge, Morris counter has not been considered for data aggregation under differential privacy regime before. Nevertheless, there are a few very recent papers (probably all were written at the same time) presenting privacy preserving protocols that use Flajolet—Martin sketch as a building block. We independently concerned more general concept of MaxGeo counter, which is a core of this sketch or HyperLogLog sketch, however it can be used in other arrangements as well.

In all the cases the conclusion is positive in the sense that the protocol itself provides some level of DP without adding extra randomness.

- In [95], authors introduce its differentially private version of Flajolet—Martin sketch via the same trick (adding artificial utilities) and provide its accuracy, when used to count the number of elements in a multisets. Accidentally, a proof of basic theorem from [95] uses incorrect argument (inappropriate utilization of Hoeffding's inequality), so it is difficult to compare the results precisely. Nevertheless the overlap of results between our paper and [95] is only partial.

- In [19], authors also consider Flajolet—Martin sketch as a subroutine. After a careful analysis, they show that it is asymptotically $(\varepsilon, \delta = negl(\lambda))$-differentially private (with respect to the numbers of different elements), when the number of elements counted by the mechanism is at least $8K\lambda \max(\frac{1}{\varepsilon}, 1)$, where K is some accuracy parameter, λ is some security parameter and $negl(x)$ is some negligible function of argument x (Theorem 4.2 in [19]). Nevertheless the analysis does not explain how to choose parameters K and λ in order to obtain (ε, δ)-differential privacy for a given ε and δ parameters. Moreover, a consideration of asymptotic behaviour (with respect to the number of unique elements n) is not relevant, when the hash function restricts the possible result to the size bounded by its domain. Our analysis of MaxGeo counter provides exact (non-asymptotic) dependence between n and parameters ε and δ.

- In [59], authors deeply analyze Flajolet—Martin sketch, and entail an algorithm, which is able to determine a minimal number of unique elements needed to guarantee (almost) (ε, δ)-differential privacy. The algorithm is quite complicated, but very efficient in terms of time of execution. Nevertheless, the authors of [59] actually proved conditions which look quite similar to (ε, δ)-differential privacy (or rather Fact 4.3.1), but in fact, these conditions are slightly weaker (in Sec. 5.2.2). Again this oversight makes it hard to compare the results from [59] with our contribution for MaxGeo counter.

Unsurprisingly one of the main applications of approximate counter is to compute a size of a database or its specific subset. A set of such applications can be found in [46]. In [104] the authors use Morris Counter for online, probabilistic and space efficient counting over streams of fixed, finite length. Authors of [24] proposed an application of a system of Morris Counters for flash memory devices. Another application, presented in [30], is a revisited version of Morris Counter designed for binary floating-point numbers. In [55] Morris Counter is used in a well-known problem of counting the frequency moments of long data streams. The authors of [34] focused on making probabilistic counters scalable and accurate in concurrent setting. Paper on probabilistic counters in hardware can be found in [91].

In random graphs theory, Morris Counter is usually connected to greedy structures. For instance, in an arrangement of a random labelled graph in Gilbert model $G(n,p)$, it is possible to construct a greedy stable set S_n, which size has the same distribution as Morris Counter M_n of the base $a = (1-p)^{-1}$ (see e.g. [48] or [16] for fundamentals of random graph theory).

There are several birth processes[7] that are quite similar to Morris Counter, which are applicable in variety of disciplines like biology, physics or theory of random graphs. Short descriptions of such examples can be found in [29].

When talking about probabilistic counters, it worth to mention about Bloom filters [15] – the space efficient probabilistic data structures, which are the representations of sets. There exists a counter which approximates the number of elements represented by a given Bloom filter [98].

Other common examples of probabilistic counters are F_p counters [3], [63], which approximate the p-th moments of frequencies of occurrences of different elements in the database.

One may note that all the mentioned probabilistic counters have equivalent versions which are are consistent with Definition 1.3.1.

4.8 Summary

4.8.1 Our contribution

The main contribution of this chapter is as follows:

- We prove that the Morris Counter satisfies $(L(n), 0.00033)$-Differential Privacy property (Definition 4.2.2 is provided in Section 4.2), where $L(n) = -\ln\left(1-\frac{16}{n}\right) \approx \frac{16}{n}$ (Theorem 4.5.1 in Section 4.5). In Observation 2 we show that constant 16 cannot be improved.

- We prove that MaxGeo Counter satisfies (ε, δ)-Differential Privacy property if a number of incrementation requests n (see Section 1.3.4 for definition) is at least $\dfrac{\ln(\delta)}{\ln\left(1-2^{-l_\varepsilon}\right)}$, where $l_\varepsilon = \left\lceil \log\left(\frac{e^\varepsilon}{e^{-\varepsilon}-1}\right)\right\rceil$ (Theorem 4.5.5 in Section 4.5).

- We construct a privacy-preserving distributed survey protocol based on probabilistic counters in Section 4.6 and compare it with Laplace method, which is the actual state-of-the-art and does not use probabilistic counters.

In this chapter we have investigated probabilistic counters from privacy-protection perspective. We have shown that both Morris Counter and MaxGeo Counter have differential privacy guarantees inherently from the mechanism itself, provided that there is at least a small, fixed number of incrementation requests. Otherwise the counter has too low value and, intuitively, the result

[7] Roughly speaking, birth processes either remain unchanged, increments or decrements the value by 1 at the time.

is not randomized enough. We have also shown, that constant in our Morris Counter result cannot be improved further.

4.8.2 Conclusions and Future Work

We have shown how to perform data aggregation, namely a distributed survey, in a privacy-preserving manner by using probabilistic counters. Note that the security model was somewhat optimistic. Unfortunately, in such setting there is little incentive to use them, other than the situation when we already have them deployed and working as aggregators due to e.g. memory-efficiency requirements. This would, however, change tremendously if we could weaken these assumptions. This seems to be a promising way to continue our research from this dissertation. Namely, we focused on privacy here, and are still missing the ability to weaken the security assumptions and allow the adversary to extract information from channels between users and the aggregator. That would put us in the, so called, Local Model, where each user is responsible for the data randomization. However, such approach require us to be able to perform probabilistic counter in an oblivious manner which, to the best of our knowledge, was not explored before.

In Section 4.3.1 we have mentioned that there exists a concept of more general Morris Counter (for bases $a > 1$). Analysis of privacy properties of such variants of Morris Counters and various probabilistic counters presented for example in [30], [49] may also be promising direction of further research.

Morris Counter and MaxGeo Counter are considered as the most popular counters. However, results of this chapter shed a new light on the properties of probabilistic counter in general. There is a possibility to provide analogous differential privacy properties for other probabilistic counters. Moreover, this chapter enables to provide differentially private algorithms for other applications, especially those, which base on Morris or MaxGeo counter.

Another interesting direction of further work may be k-DP property of Morris and MaxGeo counters. This approach generalizes the described DP property to pairs of databases that are not only identical or neighbouring, but can differ even in at most k rows.

Appendix A

Optimal Distributions for NALEA

A.1 Optimal distributions for 3 devices

According, to formula (2.10), we may obtain few first elements of the sequence $(q_L(3))_L$ (see Table A.1). One can see, that $(q_L(3))_L$ at first rise harshly, and

$a\backslash b$	1	2	3	4	5
0	0	0.66666...	0.78260...	0.83768...	0.87021...
1	0.89177...	0.90713...	0.91865...	0.92761...	0.93478...
2	0.94065...	0.94554...	0.94969...	0.95325...	0.95633...
3	0.95904...	0.96142...	0.96355...	0.96545...	0.96716...
4	0.96871...	0.97012...	0.97141...	0.97259...	0.97367...
5	0.97468...	0.97561...	0.97647...	0.97728...	0.97803...

Table A.1: Elements of a sequence $(q_{5a+b}(3))_{5a+b}$, where $a \in [0:5]$ and $b \in [5]$.

further ascends very slowly.

Basing on Table A.1, one can obtain for example $\bar{p}(3)^{(10)}$ or $\bar{p}(3)^{(20)}$ which we present in the forms of tables A.2 and A.3. In both we can see that the masses of atoms decrease and in Table A.3 we may realize that the values begins to stabilize the descent. One may also compare the appropriate masses in two tables and see that, for instance $p_4(3)^{(10)} \approx 0.09$ and $p_4(3)^{(20)} \approx 0.058$, hence the values significantly differ.

A.2 Approximations of the optimal solutions

Remark that Eq. (2.10) is quite unwieldy in practise. Indeed, we can obtain $q_i(n)$ recursively, however the computations are relatively long and memory requirements are relatively big, when we want to obtain precise results. Notice

$a\backslash b$	1	2	3	4	5
0	0.24508...	0.12254...	0.10211...	0.091018...	0.083631...
1	0.078204...	0.073972...	0.070538...	0.067669...	0.065219...

Table A.2: Elements of a sequence $(p_{5a+b}(3)^{(10)})_{5a+b}$, where $a \in [0:1]$ and $b \in [5]$.

$a\backslash b$	1	2	3	4	5
0	0.15656...	0.078281...	0.065234...	0.058144...	0.053425...
1	0.049958...	0.047255...	0.045060...	0.043228...	0.041663...
2	0.040304...	0.039108...	0.038044...	0.037087...	0.036220...
3	0.035429...	0.034704...	0.034034...	0.033414...	0.032837...

Table A.3: Elements of a sequence $(p_{5a+b}(3)^{(20)})_{5a+b}$, where $a \in [0:3]$ and $b \in [5]$.

that one have to remember all the values of $q_i(n)$ to provide $p_j(n)^{(L)}$. Therefore one can accept an eventual loss of precision by utilizing approximated distribution in order to speed up the computations.

We would like to compare different approximations of optimal distributions. In order to do so, for $k < L$, let us introduce an estimator

$$\Lambda_j(k,n,L) = \begin{cases} \frac{2}{nj}\left(\frac{j}{L}\right)^{\frac{2}{n}} & \text{for } j \in [k+1:L]\ , \\ \dfrac{1 - \sum\limits_{i=k+1}^{L} \frac{2}{ni}\left(\frac{i}{L}\right)^{\frac{2}{n}}}{\sum\limits_{i=1}^{k} a_i(n)} a_j(n) & \text{for } j \in [k]\ , \end{cases}$$

where numbers $a_j(n)$ are given by Eq. (2.7). Remark that for $j > k$, the estimator is defined as in Eq. (2.16). As announced in Section 2.5, the approximation of the atoms (2.16) is rather more precise for bigger values of j. Moreover, note that (2.16) is quite simple to calculate, however it does not define the distribution, since the sum of all approximated atoms usually does not equal 1. However, as one can easily see, for any $k \in [L]$, $\sum\limits_{j=1}^{L} \Lambda_j(k,n,L) = 1$, so every $(\Lambda_j(k,n,L))_{j=1}^{L}$ defines some distribution on $[L]$.

A.2.1 Numerical results

We are going to compare approximations $\Lambda(k,n,L)$ aforementioned before with the optimal distribution $\bar{p}(n)^{(L)}$ for some arbitrary chosen n and L. First, we will provide some numerical results for simple examples of such the distributions and further we are going to utilize two distance measures on the simplex Sim_L of probability distributions on $[L]$: total variation distance and Kullback—Leibler divergence (see Section 1.3.13 for definitions and properties).

$i\backslash D$	$p_i(10)^{(8)}$	$\Lambda_i(1,10,8)$	$\Lambda_i(2,10,8)$	$\Lambda_i(3,10,8)$	$\Lambda_i(4,10,8)$
1	0.71050...	0.705196...	0.70288...	0.70426...	0.70587...
2	0.078945...	0.075785...	0.078098...	0.078251...	0.078430...
3	0.053733...	0.054791...	0.054791...	0.053261...	0.053382...
4	0.041890...	0.043527...	0.043527...	0.043527...	0.041617...
5	0.034776...	0.036411...	0.036411...	0.036411...	0.036411...
6	0.029955...	0.031469...	0.031469...	0.031469...	0.031469...
7	0.02644...	0.027818...	0.027818...	0.027818...	0.027818...
8	0.023751...	0.025	0.025	0.025	0.025

Table A.4: Numerical comparison of the atoms of distributions on [8], for $n = 8$.

D	$\Pr[S_D]$	$\|. - \bar{p}\|_{TV}$	$D_{KL}(.\|\bar{p})$
$\bar{p}(10)^{(8)}$	0.805462...	0	0
$\bar{\Lambda}(1,10,8)$	0.80516...	0.00846963...	0.000268871...
$\bar{\Lambda}(2,10,8)$	0.80517...	0.00846963...	0.000230498...
$\bar{\Lambda}(3,10,8)$	0.80519...	0.00741141...	0.000207217...
$\bar{\Lambda}(4,10,8)$	0.80523...	0.00577422...	0.000161841...

Table A.5: Comparison of: probabilities of successful leader election, total variation distances and Kullback—Leibler divergences.
D states for a distribution, S_D is an event of a success of leader election algorithm in urn model, according to the distribution D.

As one can realize in Table A.4, the bigger the first parameter of Λ, the more precise is the approximation. Also, even, for the smallest of these parameters, the masses of atoms do not differ significantly. Moreover note, that in practise, $\Lambda(1,n,L)$ or even $\Lambda(2,n,L)$ are very easy to obtain.

Recall that according to a remark from the end of Section 2.5.6, there is no need to use more than $\lceil -\lg\varepsilon\rceil + 1$ bits in order to provide $(1-\varepsilon)$-reliable non-anonymous LEA. Therefore parameter $L = 8$ is appropriate for $\varepsilon \geqslant 0.25$. Thence, in Table A.5, we have obtained the probability of success of at least 0.75, when using $\bar{p}(10)^{(8)}$ distribution. Note that all the approximations provided in Table A.5 have relatively close probability of success to the optimal one, what affirms the design of the estimators. The same conclusion we may draw, when we take a closer look on columns with measures of similarity of distributions.

In Table A.6 and Table A.7 we presented comparisons of exact optimal distributions and the second estimator $\Lambda(2,n,L)$, for $n \in [3:6]$ and $L \geqslant 2$. When we take a closer look, we can realize that, for small n, the estimation is more efficient than for bigger ones (like in Table A.4). However, let us notice that the range of L in Table A.7 is shorter. This is due to time and memory complexity of execution of the exact distributions for bigger L. This clearly affirms the introduction of approximated distributions.

$L\backslash D$	$\bar{p}(3)^{(L)}$	$\bar{\Lambda}(2,3,L)$	$\bar{p}(4)^{(L)}$	$\bar{\Lambda}(2,4,L)$
2	0.44444...	0.44444...	0.421875	0.421875
3	0.61247...	0.61240...	0.58938...	0.58925...
4	0.70172...	0.70165...	0.68055...	0.68041...
5	0.75727...	0.75721...	0.73819...	0.73805...
6	0.79525...	0.79520...	0.77801...	0.77789...
7	0.82289...	0.82284...	0.80722...	0.80711...
8	0.84392...	0.84387...	0.82957...	0.82948...
9	0.86045...	0.86042...	0.84725...	0.84716...
10	0.87381...	0.87378...	0.86158...	0.86150...
11	0.88482...	0.88479...	0.87343...	0.87336...
12	0.89406...	0.89403...	0.88340...	0.88334...
13	0.90192...	0.90189...	0.89191...	0.89186...
14	0.90869...	0.90866...	0.89926...	0.89921...
15	0.91458...	0.91456...	0.90566...	0.90562...
16	0.91976...	0.91974...	0.91130...	0.91126...

Table A.6: Comparison of success probabilities $\Pr[S_D]$ in leader elections according to optimal distributions and theirs second approximations for $n \in \{3, 4\}$ and $L \in [2 : 16]$.

$L\backslash D$	$\bar{p}(5)^{(L)}$	$\bar{\Lambda}(2,5,L)$	$\bar{p}(6)^{(L)}$	$\bar{\Lambda}(2,6,L)$
2	0.4096	0.4096	0.40187...	0.40187...
3	0.57655...	0.57638...	0.56837...	0.56818...
4	0.66864...	0.66844...	0.66100...	0.66076...
5	0.72735...	0.72715...	0.72036...	0.72012...
6	0.76816...	0.76798...	0.76178...	0.76155...
7	0.79822...	0.79805...	0.79237...	0.79217...
8	0.82130...	0.82116...	0.81592...	0.81574...
9	0.83961...	0.83948...	0.83463...	0.83446...
10	0.85448...	0.85437...	0.84985...	0.84970...
11	0.86681...	0.86671...	0.86248...	0.86234...
12	0.87720...	0.87711...	0.87314...	0.87301...
13	0.88607...	0.88599...	0.88225...	0.88214...

Table A.7: Comparison of success probabilities $\Pr[S_D]$ in leader elections according to optimal distributions and theirs second approximations for $n \in \{5, 6\}$ and $L \in [2 : 13]$.

Appendix B

The proof of Theorem 2.8.1 and remarks

This appendix is dedicated mainly to the proof of Theorem 2.8.1 and is divided in several parts. We commence with several useful definitions and properties of random variables. Further we provide some observations, which describe necessary conditions that have to be satisfied in order to provide adequate parameters of reliable GeoGLE algorithm. After the preparation, we utilize the aforementioned observations to show that the parameters provided in Theorem 2.8.1 enables to execute $(1-\varepsilon)$-reliable GeoGLE algorithm in a case of election amongst exactly N devices. In the latter part, we prove that these parameters are also appropriate for $(1-\varepsilon)$-reliable GeoGLE algorithm, whenever the number of nodes in the network is $n \leqslant N$.

In Section B.5 we will consider how much can we exceed the capacity of the network when using one bit of memory more to preserve the $(1-\varepsilon)$-reliability.

B.1 Collision Probability in GeoGLE Algorithm

B.1.1 Definitions and crucial properties

In Section 2.7.2, we have analyzed the faultiness of using urn model according to $\mathrm{Geo}\,(p)$ distribution with infinite support. Thence we do not use directly a geometric distribution like in [64]. Nevertheless, we can take advantage of some of the results for such the model in an analysis of LEA with restricted version of geometric distribution. Hereinafter, we assume that: $G_i \sim \mathrm{Geo}\,(p)$ are i.i.d. for $i \in [n]$, $q = 1-p$, L is the maximal value of the restricted random variable $\mathrm{Geo}\,(p, L)$ and we introduce $\lambda := (n-1)q^L$.[1]

Let us define two approximations of p parameter: $\hat{p} := \frac{p(2-p)(3-5p)}{6(1-p)}$ and $\tilde{p} := \frac{p}{(1-p)} + \frac{2p^2}{3}\left(1 + \frac{p}{2(1-p)}\right)$. It worth to mention that $\hat{p} \sim p \sim \tilde{p}$ as $p \to 0^+$.

[1] If the value of n is not clear from the context, then we will write λ_n for disambiguation.

Remark that

$$\tilde{p} = p - \frac{7}{6}p^2 - \frac{1}{3}p^3 + \mathrm{O}\left(p^4\right)$$

and

$$\hat{p} = p + \frac{5}{3}p^2 + \frac{4}{3}p^3 + \mathrm{O}\left(p^4\right) \ ,$$

as $p \to 0^+$. The beneath lemma shows boundaries for $W_{\geqslant 2,n,p}$ (and consequently, for $W_{1,n,p}$):

Lemma 15. *Let $n \in \mathbb{N}$ and $p \in \left(0, \frac{2}{3}\right)$. Then $\hat{p} < 2W_{\geqslant 2,n,p} < \tilde{p}$.*

Proof. Commence with the upper bound for $W_{1,n,p}$, using Theorem 1.3.3 and inequalities (1.6):

$$\begin{aligned} W_{1,n,p} &< -\frac{p}{\ln(1-p)} - \frac{p\ln(1-p)}{3} && \text{(B.1)} \\ &< \frac{p}{p+\frac{p^2}{2}} + \frac{p}{3}\left(p + \frac{p^2}{2(1-p)}\right) = \frac{1}{1+\frac{p}{2}} + \frac{p^2}{3} + \frac{p^3}{6(1-p)} \\ &< 1 - \frac{p}{2} + \frac{7p^2}{12} + \frac{p^3}{6(1-p)} = 1 - \frac{p(2-p)(3-5p)}{12(1-p)} = 1 - \frac{\hat{p}}{2} \ . && \text{(B.2)} \end{aligned}$$

Moreover, again from Theorem 1.3.3 and inequalities (1.6)

$$\begin{aligned} W_{1,n,p} &> -\frac{p}{\ln(1-p)} + \frac{p\ln(1-p)}{3} && \text{(B.3)} \\ &> \frac{1}{1+\frac{p}{2(1-p)}} - \frac{p^2}{3}\left(1 + \frac{p}{2(1-p)}\right) \\ &\overset{p<\frac{2}{3}}{>} 1 - \frac{p}{2(1-p)} - \frac{p^2}{3}\left(1 + \frac{p}{2(1-p)}\right) = 1 - \frac{\tilde{p}}{2} \ . \end{aligned}$$

By the complements of events we abruptly procure the thesis. □

We will utilize some of the above inequalities to find an upper bound for a probability of the failure of GeoGLE algorithm.

Lemma 16. *Let $n \in \mathbb{N}$ and $p \in \left(0, \frac{1}{2}\right)$. Then*

$$1 - \frac{-\ln(1-p)}{2} - \frac{\ln^2(1-p)}{6} < W_{1,n,p} < 1 - \frac{-\ln(1-p)}{2} + \frac{\ln^2(1-p)}{2} \ .$$

Proof. For $x \in (0,\infty)$ and $\odot \in \{+,-\}^2$, let us define the following auxiliary function:

$$F_{\odot}(x) = \frac{1-e^{-x}}{x}\left(1 \odot \frac{x^2}{3}\right) \ .^3 \qquad \text{(B.4)}$$

[2]The operation $\odot$ should be interpreted polymorphically. Both as a unary operation, which prefixes either plus or minus sign to a real number and also as a binary operator of either addition or subtraction of real numbers.

[3]Remark that the first factor of $F_{\odot}(x)$ is the inversion of a generating function of Bernoulli numbers (compare with Eq. (1.11) in Section 1.2.13).

The series representations of the above two functions will be helpful in approximations of success probability in the further part of the proof. Note that from Eq. (1.4), $F_{\odot}(x)$ and both its factors have Maclaurin series representations, so all theirs derivatives have limits at $x = 0$. Moreover, realize that $\left(1 \odot \frac{x^2}{3}\right)' = \odot\frac{2x}{3}$, $\left(1 \odot \frac{x^2}{3}\right)'' = \odot\frac{2}{3}$ and $\left(1 \odot \frac{x^2}{3}\right)^{(k)} = 0$ for $k \in \mathbb{N}\backslash\{1,2\}$. Hence, according to formula (1.2),

$$\lim_{x\to 0^+} F_{\odot}{}^{(n)}(x) = \lim_{x\to 0^+} \sum_{k=0}^{2} [\![n \geqslant k]\!] \binom{n}{k} \left(1 \odot \frac{x^2}{3}\right)^{(k)} \left(\frac{1-e^{-x}}{x}\right)^{(n-k)} .^{4} \quad \text{(B.5)}$$

Moreover, realize that $\left(1 \odot \frac{x^2}{3}\right)\Big|_{x=0} = 1$ and $\left(1 \odot \frac{x^2}{3}\right)'\Big|_{x=0} = 0$. From Eq. (1.4) we know that

$$\frac{1-e^{-x}}{x} = \sum_{n=0}^{\infty} \frac{(-x)^n}{(n+1)!} . \quad \text{(B.6)}$$

According to Eq. (B.6), $\left[\frac{x^n}{n!}\right] \frac{1-e^{-x}}{x} = \lim_{x\to 0^+} \left(\frac{1-e^{-x}}{x}\right)^{(n)} = \frac{(-1)^n}{n+1}$ for every $n \in \mathbb{N}_0$. Therefore from Eq. (B.5):

$$\begin{aligned}\lim_{x\to 0^+} F_{\odot}{}^{(n)}(x) &= \lim_{x\to 0^+} \left(\frac{1-e^{-x}}{x}\right)^{(n)} \odot \lim_{x\to 0^+} \frac{2}{3}[\![n \geqslant 2]\!] \binom{n}{2} \left(\frac{1-e^{-x}}{x}\right)^{(n-2)} \\ &= \frac{(-1)^n}{n+1} \odot [\![n \geqslant 2]\!] \frac{(-1)^{n-2} n(n-1)}{3(n-1)} = \frac{(-1)^n}{n+1} \odot [\![n \geqslant 2]\!] \frac{(-1)^n n}{3}\end{aligned}$$

and therefore the Maclaurin formula for $F_{\odot}(x)$ is:

$$\begin{aligned}F_{\odot}(x) &= 1 - \frac{x}{2} + \sum_{n=2}^{\infty} \left(\frac{1}{(n+1)!} \odot \frac{1}{3(n-1)!}\right) (-x)^n \\ &= 1 - \frac{x}{2} + \sum_{n=2}^{\infty} \left(\frac{3 \odot n(n+1)}{3(n+1)!}\right) (-x)^n .\end{aligned}$$

Realize that $\lim_{n\to\infty} \sqrt[n]{\left|\frac{(-1)^n}{n+1} \odot \frac{(-1)^n n}{3}\right|} = 1$, so one may utilize Cauchy root test (see Fact 1.2.3), to show that the radii of convergence of both power series equals to 1. Note that both $\left(\frac{1}{(n+1)!}\right)_n$ and $\left(\frac{1}{3(n-1)!}\right)_n$ are decreasing sequences, which tend to 0. Now, let us consider a sequence $(c_n)_{n=2}^{\infty}$, where $c_n = \frac{3-n(n+1)}{3(n+1)!}$. First of all, we may briefly see that, for $n \in \mathbb{N}\backslash\{1\}$, $3 - n(n+1) < 0$. On the other hand

$$\frac{-c_{n+1}}{-c_n} = \frac{(n^2+3n-1)(n+1)!}{(n^2+n-3)(n+2)!} = \frac{n^2+3n-1}{n^3+3n^2-n-6} = \frac{1}{n} + \frac{6}{n(n^3+3n^2-n-6)}$$

[4] Note that according to observation about Maclaurin series representation, the right hand side makes sense.

and $n^3+3n^2-n-6 \geqslant 12$ for $n \in \mathbb{N}\backslash\{1\}$, so $(c_n)_{n=2}^{\infty}$ is non-decreasing and tends to 0. The last few facts entail that, in both cases $\odot \in \{+,-\}$,

$$\sum_{n=2}^{\infty}\left(\frac{3 \odot n(n+1)}{3(n+1)!}\right)(-x)^n$$

are Leibniz' type series for $x \in (0,1)$ (see Fact 1.2.4) and thence

$$F_+(x) = 1 - \frac{x}{2} + \frac{x^2}{2} - \frac{5x^3}{24} + \frac{23x^4}{360} + \mathrm{O}\left(x^5\right) \leqslant 1 - \frac{x}{2} + \frac{x^2}{2} \tag{B.7}$$

and

$$F_-(x) = 1 - \frac{x}{2} - \frac{x^2}{6} + \frac{x^3}{8} - \frac{17x^4}{360} + \mathrm{O}\left(x^5\right) \geqslant 1 - \frac{x}{2} - \frac{x^2}{6}\ .^5 \tag{B.8}$$

Finally, we justify the introduction of those auxiliary functions. Indeed, from the definition, $F_-(-\ln(1-p)) = -\frac{p}{\ln(1-p)} + \frac{p\ln(1-p)}{3}$ and $F_+(-\ln(1-p)) = -\frac{p}{\ln(1-p)} - \frac{p\ln(1-p)}{3}$. Moreover, from inequalities (B.3) and (B.1) in the proof of Theorem 1.3.3 and inequalities (B.8) and (B.7), we attain:

$$1 - \frac{-\ln(1-p)}{2} - \frac{\ln^2(1-p)}{6} < W_{1,n,p} < 1 - \frac{-\ln(1-p)}{2} + \frac{\ln^2(1-p)}{2}\ . \qquad \square$$

Observation 4. *Notice that $S_{n,\mathrm{Geo}(p,L)}$ can be treated as stochastically identical event to $(\mathrm{MG}_{\geqslant L}(n) < 2) \cap (W_{n,p} = 1)$.*[6] *An intuition behalf this stochastic equivalent of success is as follows — first, we need to draw a number G according to $\mathrm{Geo}\,(p)$ distribution and further we restrict the value by taking $\max(G, L)$. Then, if all the agents have drawn identities in range $[0 : L]$, then the truncation does not change values of realizations. Hence if there is exactly one maximum among them, then there is so for the truncated values. Otherwise, if there is exactly one node i, who have drawn a realization $G_i \sim \mathrm{Geo}\,(p)$, which exceeds or equals to L, then the truncation affects only agent i and the changed identifier is unique and the biggest. Conversely, if there are more such the nodes, which drew at least value L, then theirs truncations will provide L and therefore a collision.*

B.2 Constraints of accurate GeoGLE algorithm

Lemma 17. *Let $n \in \mathbb{N}\backslash\{1\}$ and $0 < \lambda < 1$. Then*

$$0 < e^{-\lambda} - \left(1 - \frac{\lambda}{n-1}\right)^{n-1} \leqslant \frac{\lambda^2}{2(n-1)}\ .$$

[5] Both $\mathrm{O}\,()$ terms are concerned as $x \to 0$.

[6] for definition of $(\mathrm{MG}_{\geqslant L}(n) < 2)$, see Section 2.9.3.

Proof. Let us observe that from Eq. (1.4) and Newton's Binomial Formula (see Fact 1.2.6), we obtain:

$$e^{-\lambda}-\left(1-\frac{\lambda}{n-1}\right)^{n-1}=\sum_{k=0}^{\infty}\frac{(-\lambda)^k}{k!}-\sum_{k=0}^{n-1}\binom{n-1}{k}\left(-\frac{\lambda}{n-1}\right)^k$$
$$=\sum_{k=2}^{n-1}\frac{(-\lambda)^k}{k!}\left(1-\frac{\prod_{i=1}^{k-1}(n-1-i)}{(n-1)^{k-1}}\right)+\sum_{k=n}^{\infty}\frac{(-\lambda)^k}{k!}\ .$$

The above function is given by the alternating power series of form $\sum_{k=2}^{n-1}(-\lambda)^k a_k+\sum_{k=n}^{\infty}(-\lambda)^k b_k$, where $a_k=\frac{1}{k!}\left(1-\prod_{i=1}^{k-1}\left(1-\frac{i}{n-1}\right)\right)$ and $b_k=\frac{1}{k!}$. Realize that the product embraced in the formula for a_k can be easily bounded by 1 or its first factor, so for $k\in[2:n-2]$, we attain:

$$\frac{a_{k+1}}{a_k}=\frac{1}{k+1}\frac{\left(1-\prod_{i=1}^{k}\left(1-\frac{i}{n-1}\right)\right)}{\left(1-\prod_{i=1}^{k-1}\left(1-\frac{i}{n-1}\right)\right)}=\frac{1}{k+1}\frac{\left(1-\left(1-\frac{k}{n-1}\right)\prod_{i=1}^{k-1}\left(1-\frac{i}{n-1}\right)\right)}{\left(1-\prod_{i=1}^{k-1}\left(1-\frac{i}{n-1}\right)\right)}$$
$$=\frac{1}{k+1}\left(1+\frac{k\prod_{i=1}^{k-1}\left(1-\frac{i}{n-1}\right)}{(n-1)\left(1-\prod_{i=1}^{k-1}\left(1-\frac{i}{n-1}\right)\right)}\right)$$
$$\leqslant\frac{1}{k+1}\left(1+\frac{k}{(n-1)\left(1-\left(1-\frac{1}{n-1}\right)\right)}\right)=1\ .$$

Obviously $b_k>b_{k+1}$ for $k\in\mathbb{N}\backslash[n-1]$, so we would like to connect the above monotonicity properties. Therefore we are going to check the ratio:

$$\frac{b_n}{a_{n-1}}=\frac{1}{n}\frac{1}{\left(1-\prod_{i=1}^{n-2}\left(1-\frac{i}{n-1}\right)\right)}\leqslant\frac{1}{n\left(1-\left(1-\frac{1}{n-1}\right)\right)}<1\ .$$

In a result, a concatenation of these sequences – $(a_k)_{k=2}^{n-1}\cup(b_l)_{l=n}^{\infty}$ – forms a decreasing sequence.[7] Therefore $e^{-\lambda}-\left(1-\frac{\lambda}{n-1}\right)^{n-1}$ can be represented as Leibniz' type series for $0<\lambda<1$, so Fact 1.2.4 provides the bounds from the formulation of this lemma. □

[7]Remark that $\lambda^2 a_2=\frac{\lambda^2}{2(n-1)}$ (when $n=2$, the bound is given by $(b_k)_{k=2}^{\infty}$ sequence, so the (strict) upper bound may be given by $\lambda^2 b_2=\frac{\lambda^2}{2}=\frac{\lambda^2}{2(n-1)}$); see Section 1.2.1 for definitions of concatenation.

In here we introduce assumptions commonly used in the further part of this appendix:

Asm.1 $n \in \mathbb{N}\backslash\{1\}$, $L \in \mathbb{N}$ and $p \in (0,1)$,

Asm.2 $c \in (0,4)$ and $\varepsilon \leqslant \frac{c^2}{(c+2)^3}$,

Asm.3 $\Pr\left[S_{n,\mathrm{Geo}(p,L)}\right] > 1-\varepsilon$.

Lemma 18. *Let us assume (Asm.1) and (Asm.2). Then, from (Asm.3) we entail* $\lambda < \sqrt{(c+2)\varepsilon} < \frac{2}{3}$.

Proof. First, note that (Asm.2) bears $\sqrt{(c+2)\varepsilon} \leqslant \frac{c}{c+2} \overset{c<4}{<} \frac{2}{3}$.

Realize that according to Observation 4, $(\mathrm{MG}_{\geqslant L}(n) \geqslant 2)$ implies the collision in GeoGLE algorithm, hence from (Asm.3), $\Pr[\mathrm{MG}_{\geqslant L}(n) \geqslant 2] < 1 - \Pr\left[S_{n,\mathrm{Geo}(p,L)}\right] < \varepsilon$. Now, we will show that $\lambda < \frac{2}{3}$ as well. Indeed, with (Asm.2), let us define $h(c) := \frac{c^2}{(c+2)^3}$. Realize that $h'(c) = \frac{2c(c+2)^3 - 3(c+2)^2c^2}{(c+2)^6} = \frac{c(4-c)}{(c+2)^4} > 0$, so for $c \in (0,4)$, h is increasing and consequently $\varepsilon < \lim\limits_{c\to 4^-} h(c) = \frac{2}{27} = 0.(074)$. Since $\left(\left(1-\frac{\lambda}{n-1}\right)^{n-1}\right)_{n=1}^{\infty}$ is ascending sequence, we briefly see from (2.22) that

$$(1+\lambda)e^{-\lambda} > \Pr[\mathrm{MG}_{\geqslant L}(n) < 2] \ . \tag{B.9}$$

Suppose for a while that $\lambda \geqslant \frac{2}{3}$ and let $g(\lambda) := (1+\lambda)e^{-\lambda}$, which is positive for any $\lambda > 0$. Its derivative satisfies $g'(\lambda) = -\lambda e^{-\lambda} < 0$ for $\lambda > 0$. Hence $g(\lambda)$ reaches a maximum in the domain $[\frac{2}{3}, \infty)$ for $\lambda = \frac{2}{3}$. Then, from (B.10), we obtain $\varepsilon > 1 - (1+\lambda)e^{-\lambda} \geqslant 1 - \frac{5}{3}\exp\left(-\frac{2}{3}\right) = 0.144305\ldots > 0.(074)$.

Let us define next auxiliary function $f(\lambda) := \lambda^2 \frac{1-\lambda}{2}$. We will show that $f(\lambda) < \varepsilon$ (compare with (2.23)) implies that $\lambda < \sqrt{(c+2)\varepsilon}$. It is clear that $f(\lambda)$ is positive for $0 < \lambda < 1$. It is also ascending for $\lambda \in \left(0, \frac{2}{3}\right)$ since $f'(\lambda) = \frac{\lambda}{2}(2-3\lambda) > 0$. Realize that $f(\sqrt{(c+2)\varepsilon}) = \frac{c+2}{2}\varepsilon(1-\sqrt{(c+2)\varepsilon})$. Imagine that $f(\sqrt{(c+2)\varepsilon}) < \varepsilon$, what is equivalent to $1 - \frac{2}{c+2} = \frac{c}{c+2} < \sqrt{(c+2)\varepsilon}$, so finally $\varepsilon > \frac{c^2}{(c+2)^3}$, contrary to (Asm.2), what shows that $f(\lambda) < f(\sqrt{(c+2)\varepsilon})$. Both arguments are less than $\frac{2}{3}$, so $\lambda < \sqrt{(c+2)\varepsilon}$ as well. □

Observation 5. *Let us focus on the fact, that we have concluded from (Asm.2), that* $\varepsilon < \frac{2}{27}$. *Moreover, if* $\Pr[\mathrm{MG}_{\geqslant L}(n) \geqslant 2] < \varepsilon$, *then* $\varepsilon < \frac{2}{27}$ *entails that* $\lambda < \frac{2}{3}$. *In other words, if* $\lambda \geqslant \frac{2}{3}$, *then, by Observation 4, (Asm.3) cannot be fulfilled.*

Corollary 9. *If assumptions (Asm.1), (Asm.2) and (Asm.3) are fulfilled, then*

$$(1+\lambda)e^{-\lambda} > \Pr[\mathrm{MG}_{\geqslant L}(n) < 2] > (1+\lambda)\left(e^{-\lambda} - \frac{\lambda^2}{2(n-1)}\right) \ . \tag{B.10}$$

The above is a conclusion from Lemma 17 and (2.22).

Observation 6. *According to the monotonicity of function h from the proof of Lemma 18, $\varepsilon = h(c)$ has exactly one solution for $c \in (0,4)$. Therefore we can denote this root by $c(\varepsilon)$. Note that then, every $c \geqslant c(\varepsilon)$ is appropriate for (Asm.2), but the most strict bound for λ in the formulation of Lemma 17 is realized by $c(\varepsilon)$. Therefore, primarily we would like to consider only $c = c(\varepsilon)$ to get the most efficient constraints.*

Remark that according to Observation 4, and (B.10), for any $n \in [2:N]$, we obtain the following lower bound for success probability:

$$\begin{aligned}\Pr\big[S_{n,\mathrm{Geo}(p,L)}\big] &= W_{1,n,p}\Pr[\mathrm{MG}_{\geqslant L}(n) < 2] > W_{1,n,p}(1+\lambda)\left(e^{-\lambda} - \frac{\lambda^2}{2(n-1)}\right) \\ &= W_{1,n,p}(1+\lambda)e^{-\lambda} - W_{1,n,p}(1+\lambda)\frac{\lambda^2}{2(n-1)}\ . \end{aligned} \tag{B.11}$$

Lemma 19. *If (Asm.1), (Asm.2) and (Asm.3), then $p < 3\varepsilon$.*

Proof. According to (B.11) and (B.2) from the proof of Lemma 15 we get

$$1-\varepsilon < \Pr\big[S_{n,\mathrm{Geo}(p,L)}\big] < W_{1,n,p} < 1 - \frac{p(2-p)(3-5p)}{12(1-p)}\ ,$$

which shows that $12\varepsilon > \frac{p(2-p)(3-5p)}{1-p}$. Assume for a while that $p \geqslant 3\varepsilon$. Then $12\varepsilon > \frac{3\varepsilon(2-3\varepsilon)(3-15\varepsilon)}{1-3\varepsilon}$ and consequently, from Observation 5:

$$\begin{aligned} 4 &> \frac{6-39\varepsilon+45\varepsilon^2}{1-3\varepsilon} = 6 + \frac{-21\varepsilon+45\varepsilon^2}{1-3\varepsilon} = 6 - 21\varepsilon + \frac{-18\varepsilon^2}{1-3\varepsilon} \\ &> 6 - \frac{14}{9} - \frac{8}{63} > 6 - \frac{106}{63} > 4\ , \end{aligned}$$

what shows the contradiction. □

In order to provide the appropriate parameters of the algorithm we would like to omit somehow a problematic part $\Lambda_n := -W_{1,n,p}(1+\lambda)\frac{\lambda^2}{2(n-1)}$ in the lower bound for the success probability in (B.11). Nevertheless, this part is essential and cannot be omitted (see Lemma 17). It can be showed that it cannot be compensated e.g. by inequality (B.3). [8] In return, we are going to bypass this problem via the following trick — we will find an upper uniform bound C for Λ_n and we will try to fulfil the condition $\Pr\big[S_{n,\mathrm{Geo}(p,L)}\big] \geqslant 1-\varepsilon+C$ instead.

Observation 7. *With (Asm.1), (Asm.2) and (Asm.3), we may briefly see from inequality (2.23) and Lemma 18 that*

$$\frac{\varepsilon}{\lambda^2} \geqslant 1 - \frac{(n-2)(1+\lambda)}{2(n-1)} > 1 - \frac{5(n-2)}{6(n-1)} = \frac{n+4}{6n-6}\ ,$$

[8] This calculation is purely technical, do not utilize any uncommon techniques and is not significant from a point of view of further considerations, hence it will be omitted.

so consequently

$$\lambda^2 < \frac{6n-6}{n+4}\varepsilon \leqslant (n-1)\varepsilon \ .$$

Therefore from (2.23) once again and Observation 5, we may obtain:

$$(1+\lambda)\frac{\lambda^2}{2(n-1)} < \frac{1+\sqrt{\frac{6n-6}{n+4}\frac{2}{27}}}{2}\varepsilon \ , \tag{B.12}$$

what can be useful to efficiently bound the left hand side of inequality (B.12), especially for small values of n. Note that $\frac{6n-6}{n+4} < 6$ for any $n \in \mathbb{N}$ and from Lemma 18, $\lambda^2 < (c+2)\varepsilon$, where $c+2 < 6$ as well. Moreover, this bound has sense only when $\frac{6n-6}{n+4} < c+2$, or equivalently for $n \leqslant \frac{4c+14}{4-c}$. The right hand side is ascending with respect to c, so the bigger the c, the broader is the applicability of (B.12). Nevertheless, we should bear in mind Observation 6, so primarily we would like c parameter to be the least possible.

Consider the beneath alternatives of (Asm.1), (Asm.2) and (Asm.3):

Asm*.1 $N \in \mathbb{N}$, $n \in [2:N]$, $L \in \mathbb{N}$ and $p \in (0,1)$,

Asm*.2 $0 < \varepsilon < \frac{2}{27}$ and $c := c(\varepsilon) \in (0,4)$ be such that $\varepsilon = \frac{c^2}{(c+2)^3}$,

Asm*.3 $\lambda_n = (n-1)(1-p)^L < \sqrt{(c+2)\varepsilon} < \frac{2}{3}$.

Beneath lemma shows the core idea of the aforementioned trick:

Lemma 20. *Assume (Asm*.1), (Asm*.2) and (Asm.3). Moreover, let us denote*

$$C(n,\varepsilon) := (1+\sqrt{(c+2)\varepsilon})\frac{(c+2)\varepsilon}{2(n-1)} \ .^{9} \tag{B.13}$$

Then

$$(\forall\, n \in [2:N])\ \Pr\big[S_{n,\mathrm{Geo}(p,L)}\big] + C(N,\varepsilon) \geqslant W_{1,n,p}(1+\lambda_n)e^{-\lambda_n} \ . \tag{B.14}$$

Proof. Recall that we treat λ as a function of parameters n, q and L. However, let as fix q and L for a while. Recall that $\lambda_n = (n-1)(1-p)^L$. First of all, realize that $\frac{\lambda_n^2}{2(n-1)} = \frac{(n-1)q^{2L}}{2}$, so it is positive and, perversely, it is ascending with respect to n. Thence, after taking into account that $W_{1,n,p} < 1$, we briefly see that from Lemma 18 we procure:

$$W_{1,n,p}(1+\lambda_n)\frac{\lambda_n^2}{2(n-1)} \leqslant (1+\sqrt{(c+2)\varepsilon})\frac{(c+2)\varepsilon}{2(N-1)} \ . \tag{B.15}$$

Now, from (B.11) and (B.15) we simply gather the property (B.14). □

Observation 8. *Assume (Asm*.1), (Asm*.2) and (Asm*.3). Realize that (B.15) is fulfilled, since (Asm*.3) interchanged Lemma 18. Therefore, from (B.13), (B.15) and Lemma 20 we conclude that*

$$\big(W_{1,n,p}(1+\lambda_n)e^{-\lambda_n} > 1-\varepsilon+C(N,\varepsilon)\big) \Rightarrow \big(\Pr\big[S_{n,\mathrm{Geo}(p,L)}\big] > 1-\varepsilon\big)\,. \tag{B.16}$$

[9] Recall that by Observation 6 and Observation 5, c is unique and is only dependent on ε, so c is not listed as an argument of C.

Remark Realize that $W_{1,n,p}(1+\lambda_n)e^{-\lambda_n} < 1$, so if $C(N,\varepsilon) \geqslant \varepsilon$, then the condition of (B.16) cannot be fulfilled. With (Asm*.2), $C(N,\varepsilon) = \frac{2c+2}{c+2}\frac{(c+2)\varepsilon}{2(N-1)} = \frac{(c+1)\varepsilon}{N-1}$, so $C(N,\varepsilon) < \varepsilon$, whenever $N > c+2$. However, note that for $N \leqslant 6$, we may use different uniform bound $\tilde{C}$, which takes Observation 7 into account instead of the general bound C. Indeed, by (B.12), we can use alternatively:

$$W_{1,n,p}(1+\lambda_n)\frac{\lambda_n^2}{2(n-1)} \leqslant \frac{1+\sqrt{\frac{6N-6}{N+4}\frac{2}{27}}}{2}\varepsilon =: \tilde{C}(N,\varepsilon)\ .$$

Note that $\tilde{C}(N,\varepsilon) < \frac{5}{6}\varepsilon$, for any $N \in \mathbb{N}$. Therefore, we are always able to guarantee

Corollary 10. $\min(C(N,\varepsilon), \tilde{C}(N,\varepsilon)) < \varepsilon$.

Remark Let us assume for a while that $\min(C(N,\varepsilon), \tilde{C}(N,\varepsilon)) \approx \frac{5}{6}\varepsilon$. Intuitively it means that we are going to provide almost $\left(1-\frac{\varepsilon}{6}\right)$-reliable leader election instead of $(1-\varepsilon)$-reliable one. According to Section 2.5.6, then we need approximately at least $\lg 6$ bits more to provide such the LEA. Hence we are rather interested in more efficient bounds $\min(C(N,\varepsilon), \tilde{C}(N,\varepsilon))$. On the other hand, we have to bear in mind, that in practise we are usually interested in big parameter N (e.g. $N = 10^9$) and small parameter ε (e.g. $\varepsilon = 10^{-10}$), which, according to (Asm*.2) gives small values of c ($c \sim \sqrt{\varepsilon}$, as $\varepsilon \to 0^+$). In such the case, $C(N,\varepsilon) \ll \varepsilon$, so then this detriment is negligible.[10]

Realize that from inequality (B.8) and Lemma 19 we get

$$W_{1,n,p} > 1 + \frac{\ln(1-p)}{2} - \frac{\ln(1-p)^2}{6} > 1 + \frac{\ln(1-p)}{2}\left(1 - \frac{\ln(1-3\varepsilon)}{3}\right)\ . \quad \text{(B.17)}$$

Thence, let us define the last factor as:

$$\tau := \left(1 - \frac{\ln(1-3\varepsilon)}{3}\right) = 1 + \varepsilon + 1.5\varepsilon^2 + \mathrm{O}\left(\varepsilon^3\right) \quad \text{(B.18)}$$

as $\varepsilon \to 0^+$.[11]

B.3 Maximal number of devices — a case $n = N$

Our goal is to find such the parameter p, which minimizes the number of needed rounds K. Naturally, then the image of the geometric draw has $L+1 = 2^K$ elements (the less L is, the less is the probability $\Pr[\mathrm{MG}_{\geqslant L}(n) < 2]$). Moreover, we would like to show that if the algorithm is accurate for $n = N$ devices, then is so for $n < N$ processors. Therefore, from now until the end of this section,

[10] Otherwise we may always assume bigger N that demanded. $C(N,\varepsilon)$ should be treated then as inefficient approximation of deficiency of unreliability of algorithm.

[11] The approximation is given by a truncation of Taylor series from Eq. (1.5)

we assume that $n = N$ and we try to establish such the parameters p and L that the GeoGLE algorithm is accurate. Moreover, we may assume that p, and consequently λ, are functions of L parameter. As long as $n = N$ is fixed, we may write that $\lambda(L) = (N-1)(1-p(L))^L$, so

$$-\frac{\ln\left(\frac{\lambda(L)}{N-1}\right)}{L} = -\ln(1-p(L)) \ . \tag{B.19}$$

A statement of Theorem 2.8.1, exactly for the number of devices equals to the capacity of the network is given as follows:

Theorem B.3.1. *Assume $N \in \mathbb{N}$, (Asm*.2) and (Asm*.3). Moreover, let*

1. $C(N,\varepsilon) = (1+\sqrt{(c+2)\varepsilon})\frac{(c+2)\varepsilon}{2(N-1)}$
2. $\tilde{C}(N,\varepsilon) = \frac{1+\sqrt{\frac{6N-6}{N+4}\frac{2}{27}}}{2}\varepsilon$,
3. $\tilde{\varepsilon} = \varepsilon - \min(C(N,\varepsilon), \tilde{C}(N,\varepsilon))$,
4. $\tau = 1 - \frac{\ln(1-3\varepsilon)}{3}$,
5. $K = \left\lceil \lg\left(1 - \frac{W_{-1}\left(-\frac{2\tilde{\varepsilon}}{e^2(N-1)^2}\right)}{4\tilde{\varepsilon}}\right)\right\rceil$,
6. $L = 2^K - 1$,
7. $p = 1 - \left(\frac{-2W_0\left(-\sqrt{\frac{\tau}{8L}}\right)}{N-1}\right)^{\frac{1}{L}}$.

Then $\Pr\left[S_{N,\mathrm{Geo}(p,L)}\right] \geqslant 1-\varepsilon$.

Proof. Realize that (Asm*.1) is satisfied.[12] Now, we are going to consider the right hand side of the inequality (B.14) for $n = N$, then apply (B.17) and substitute (B.18) and (B.19) in order to obtain:

$$\begin{aligned} W_{1,N,p}(1+\lambda)e^{-\lambda} &> \left(1 + \tau\frac{\ln\left(\frac{\lambda}{N-1}\right)}{2L}\right)(1+\lambda)e^{-\lambda} \\ &> \tau\frac{\ln\left(\frac{\lambda}{N-1}\right)}{2L} + (1+\lambda)e^{-\lambda} \ . \end{aligned} \tag{B.20}$$

[12] Note that $K \in \mathbb{N}$, what follows easily from the fact that $W_{-1}(x) < -1$ for $x \in (-\frac{1}{e}, 0)$ (see Section 1.2.12).

Therefore let us define

$$s(\lambda; L) := \tau \frac{\ln\left(\frac{\lambda}{N-1}\right)}{2L} + (1+\lambda)e^{-\lambda} \ . \tag{B.21}$$

First of all, realize that we think of s as the function of argument λ and a parameter L. Note that, if we do not change λ and increase L, then both the probability of success and $s(\lambda; L)$ increase. However, first of all, we should minimize the number of bits K. Therefore we should always choose L of the form $2^K - 1$ and we are interested in maximizing the function s with respect to λ, for some fixed unknown parameter L, which will be established later. This way we will optimize the difference between our approximation and the actual probability of success. Let us compute the following derivative:

$$\frac{\partial}{\partial\lambda} s(\lambda; L) = \frac{\tau}{2\lambda L} + e^{-\lambda} - (1+\lambda)e^{-\lambda} = \frac{\tau}{2\lambda L} - \lambda e^{-\lambda} \ .$$

If $\frac{\partial}{\partial\lambda} s(\lambda; L) = 0$, then $\frac{\tau}{2L} = \lambda^2 e^{-\lambda}$, or equivalently: $-\frac{\sqrt{\tau}}{\sqrt{8L}} = -\frac{\lambda}{2} e^{-\frac{\lambda}{2}}$. Using W-Lambert function, $\lambda(L) = -2W\left(-\frac{\sqrt{\tau}}{\sqrt{8L}}\right)$. The argument of W-Lambert multi-function is negative, so there are two possible solutions of this equation. Note that since $\varepsilon < \frac{2}{27}$, then $K \geqslant -\left\lceil (\lg\left(\frac{27}{2}\right)\right\rceil = 4$, so $L \geqslant 2^4 - 1 = 15$. Moreover, realize that from Eq. (B.19) emerges:

$$\begin{aligned}\lambda &= (N-1)\left(1 - \frac{\ln\left((N-1)\sqrt{2L}\right)}{L}\right)^L \\ &\leqslant (N-1)\exp\left(-\ln\left((N-1)\sqrt{2L}\right)\right) = \frac{1}{\sqrt{2L}} \leqslant \frac{1}{\sqrt{30}} < \frac{2}{3} \ .\end{aligned}$$

Thence we know that $W\left(-\frac{\sqrt{\tau}}{\sqrt{8L}}\right) > -\frac{1}{3}$, so we need to use the positive branch of W-Lambert function (see Section 1.2.12). In particular, from formula (1.8) we get

$$\begin{aligned}W_0\left(-\frac{\sqrt{\tau}}{\sqrt{8L}}\right) &= \sum_{i=1}^{\infty} (-1)^{i-1} \frac{i^{i-2}}{(i-1)!} \left(-\frac{\sqrt{\tau}}{\sqrt{8L}}\right)^i = -\sum_{i=1}^{\infty} \frac{i^{i-2}}{(i-1)!} \left(\sqrt{\frac{\tau}{8L}}\right)^i \\ &= -\sqrt{\frac{\tau}{8L}} - \frac{\tau}{8L} - \frac{3}{2}\left(\frac{\tau}{8L}\right)^{\frac{3}{2}} - \frac{8}{3}\left(\frac{\tau}{8L}\right)^2 + \mathrm{O}\left(L^{-\frac{5}{2}}\right) < -\sqrt{\frac{\tau}{8L}} \ ,\end{aligned}$$

as $L \to \infty$.[13] The beneath calculation shows the concavity of s with respect to λ:

$$\frac{\partial^2}{\partial\lambda^2} s(\lambda; L) = -\frac{\tau}{2\lambda^2 L} + \lambda e^{-\lambda} - e^{-\lambda} = -\frac{\tau}{2\lambda^2 L} - (1-\lambda)e^{-\lambda} < 0 \ ,$$

[13] Note that $\tau \to 1$ as $\varepsilon \to 0^+$ according to (B.18). The inequality can be straightforwardly obtained from the concavity of W_0 and the derivative $W_0'(0) = 1$. Note that according to Section 2.5.6 or Section 2.6.4 and observation that $L = 2^K - 1$, we can conclude that $L + 1 \geqslant \frac{1}{\varepsilon}$, so when ε is close to 0, then the approximation is relatively efficient.

because we consider only $0 < \lambda < \frac{2}{3}$.

Hence the maximal value of s is obtained for $\lambda(L) = -2W_0\left(-\frac{\sqrt{\tau}}{\sqrt{8L}}\right) > \sqrt{\frac{\tau}{2L}}$ and

$$\begin{aligned} W_{1,N,p}(1+\lambda)e^{-\lambda} &> s\left(-2W_0\left(-\frac{\sqrt{\tau}}{\sqrt{2L}};L\right)\right) \geqslant s\left(\frac{\sqrt{\tau}}{\sqrt{2L}};L\right) \\ &= \frac{\tau \ln\left(\frac{\sqrt{\tau}}{(N-1)\sqrt{2L}}\right)}{2L} + \left(1+\frac{\sqrt{\tau}}{\sqrt{2L}}\right)e^{-\frac{\sqrt{\tau}}{\sqrt{2L}}} \\ &> -\tau\frac{\ln\left(\frac{(N-1)\sqrt{2L}}{\sqrt{\tau}}\right)}{2L} + \left(1+\frac{\sqrt{\tau}}{\sqrt{2L}}\right)\left(1-\frac{\sqrt{\tau}}{\sqrt{2L}}\right) \\ &= 1-\tau\frac{\ln\left(\frac{(N-1)\sqrt{2L}}{\sqrt{\tau}}\right)}{2L} - \frac{\tau}{2L}\ . \end{aligned} \tag{B.22}$$

We desire to keep the probability of success above $1-\varepsilon$, so according to Observation 8, we want to solve the following inequality:

$$\tilde{\varepsilon} \geqslant \tau\frac{\ln\left(\frac{(N-1)\sqrt{2L}}{\sqrt{\tau}}\right)}{2L} + \frac{\tau}{2L} = \frac{\tau}{2L}\ln\left(\frac{e(N-1)\sqrt{2L}}{\sqrt{\tau}}\right) \tag{B.23}$$

or equivalently

$$-\frac{\tau}{e^2(N-1)^2 2L}\ln\left(\frac{e^2(N-1)^2 2L}{\tau}\right) \geqslant \frac{-2\tilde{\varepsilon}}{e^2(N-1)^2} =: H(\tilde{\varepsilon})\ .^{14}$$

Let $\ell(L) := -\ln\left(\frac{e^2(N-1)^2 2L}{\tau}\right)$ for a while. Then, we can rewrite the above inequality as $\ell(L)e^{\ell(L)} \geqslant H(\tilde{\varepsilon})$. Realize that $\ell(L) < -1$, so $\ell(L)$ can be constrained in terms of H function in a language of the alternative branch of W-Lambert multi-function — $\ell(L) \leqslant W_{-1}(H(\tilde{\varepsilon})) < -1$. We exponentiate both sides of inequality, then multiply by $W_{-1}(H(\tilde{\varepsilon}))$ to find out that

$$H(\tilde{\varepsilon}) = W_{-1}\left(H(\tilde{\varepsilon})\right)e^{W_{-1}(H(\tilde{\varepsilon}))} \leqslant W_{-1}\left(H(\tilde{\varepsilon})\right)e^{\ell(L)}$$

or explicitly

$$-\frac{2\tilde{\varepsilon}}{e^2(N-1)^2} \leqslant W_{-1}\left(-\frac{2\tilde{\varepsilon}}{e^2(N-1)^2}\right)\frac{\tau}{(N-1)^2e^2 2L}\ ,$$

so $L \geqslant -\dfrac{\tau W_{-1}\left(-\frac{2\tilde{\varepsilon}}{e^2(N-1)^2}\right)}{4\tilde{\varepsilon}}$ and therefore:

$$2^K \geqslant 1 - \frac{\tau W_{-1}\left(-\frac{2\tilde{\varepsilon}}{(N-1)^2e^2}\right)}{4\tilde{\varepsilon}} \overset{(1.10)}{\approx} 1 + \tau\frac{2+2\ln(N-1)-\ln(2\tilde{\varepsilon})}{4\tilde{\varepsilon}}\ .$$

[14] H does not treat N as an argument, since it is only considered in this section (where N is fixed).

Hence K should be attained as the smallest natural number, which fulfils the above inequality. Then we put $L = 2^K - 1$. Moreover, from (B.19), we obtain:

$$p = 1 - \left(\frac{\lambda(L)}{(N-1)}\right)^{\frac{1}{L}} = 1 - \left(\frac{-2W_0\left(-\sqrt{\frac{\tau}{8L}}\right)}{N-1}\right)^{\frac{1}{L}} .$$

Observation 8 ends the proof of Theorem B.3.1. □

B.4 Proof of Theorem 2.8.1 for $n < N$

In this part we show that parameters established in Section B.3, in Theorem B.3.1 are also appropriate for $n < N$ and in conclusion, Theorem 2.8.1 is satisfied.

Proof of Theorem 2.8.1. From (B.17) and Observation 4, we obtain

$$\begin{aligned}\Pr\left[S_{n,\mathrm{Geo}(p,L)}\right] &= W_{1,n,p}\Pr[\mathrm{MG}_{\geqslant L}(n) < 2] \\ &> \left(1 - \frac{-\ln(1-p)}{2} - \frac{\ln(1-p)^2}{6}\right)\Pr[\mathrm{MG}_{\geqslant L}(n) < 2] \ . \end{aligned} \tag{B.24}$$

Thence let us define

$$\chi_{n,p,L} := \left(1 - \frac{-\ln(1-p)}{2} - \frac{\ln(1-p)^2}{6}\right)\Pr[\mathrm{MG}_{\geqslant L}(n) < 2]$$

and $\xi_k = \left(1 - \frac{-\ln(1-p)}{2} - \frac{\ln(1-p)^2}{6}\right)\left(1-q^L\right)^{N-k-1}$ for $k \in [N-1]$.[15] We claim that $\chi_{n,p,L}$ is a decreasing sequence with respect to n for $n \in [N]$ when p and L are fixed. Namely, from Proposition 1, Lemma 18 and Weierstrass' Product Inequality (Theorem 1.4.2), we have

$$\begin{aligned}\chi_{N,p,L} - \chi_{N-k,p,L} &= \xi_k\left[\left(1+(N-1)q^L\right)\left(1-q^L\right)^k - \left(1+(N-k-1)q^L\right)\right] \\ &\overset{\text{WPI}}{\leqslant} \xi_k[\left(1+(N-1)q^L\right)\left(1 - kq^L + \frac{k(k-1)}{2}q^{2L}\right) - \left(1+(N-k-1)q^L\right)] < \\ &kq^{2L}\xi_k\left(-(N-1) + \frac{(1+\lambda)(N-1)}{2}\right) = (N-1)kq^{2L}\xi_k\frac{\lambda-1}{2} \overset{\text{Lem.18}}{<} 0 \ . \end{aligned}$$

In Section B.3, we have already proved that

$$\chi_{N,p,L} > s(\lambda_N; L) - \min(C(N,\varepsilon), \tilde{C}(N,\varepsilon)) > 1 - \varepsilon \ .$$

Consequently, for $n \in [N]$, $\Pr\left[S_{n,\mathrm{Geo}(p,L)}\right] \geqslant \chi_{n,p,L} > \chi_{N,p,L} > 1 - \varepsilon$. □

[15] Note that ξ_k is defined in such a way, that it is not dependent on n. This shows the necessity of using (B.17) in (B.24).

Final remarks

It worth to mention that we proved that, for fixed p and L, the sequence $\chi_{n,p,L}$ is monotonic with respect to n, as $n \leqslant N$. However it does not imply that the similar fact is true for the sequence $\Pr\left[S_{n,\mathrm{Geo}(p,L)}\right]$. In Figure B.1 we present a comparison of $\Pr\left[S_{n,\mathrm{Geo}(0.012423\ldots,2^9-1)}\right]$ and $\chi_{n,0.012423\ldots,2^9-1}$ for $n \in [20]$.[16] One may realize that with $p = 0.012423\ldots$,

$$\Pr\left[S_{2,\mathrm{Geo}(p,2^9-1)}\right] < \Pr\left[S_{3,\mathrm{Geo}(p,2^9-1)}\right] > \Pr\left[S_{4,\mathrm{Geo}(p,2^9-1)}\right] \ ,$$

so indeed the probability of success does not have to be monotonic. Naturally, for all $n \in [20]$, a condition

$$\Pr\left[S_{n,\mathrm{Geo}(0.012423\ldots,2^9-1)}\right] > \chi_{n,0.012423\ldots,2^9-1}$$

is satisfied, in accordance with the proof of Theorem 2.8.1. It worth to note that the differences between those two sequences are relatively big, however it is an aftermath of quite big fluctuations of $W_{1,n,0.012423\ldots}$ (see Theorem 1.3.3).

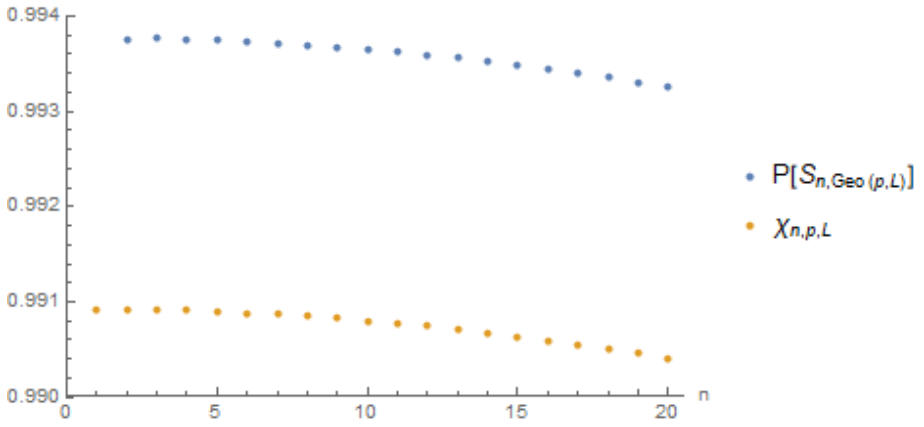

Figure B.1: Comparison of probability of success of GeoGLE algorithm according to $\mathrm{Geo}\left(0.012423\ldots,2^9-1\right)$ distribution in urn model and elements $\chi_{n,0.012423\ldots,2^9-1}$.

In Section B.2 we have showed that both $C(n,\varepsilon)$ and $\tilde{C}(n,\varepsilon)$ are increasing functions with respect to n parameter, however we do not know anything about monotonicity of $s(\lambda_n;L)$ with respect to n, so the conclusion of the correctness for $n < N$ is not straightforward.

[16] Realize that $\Pr\left[S_{1,\mathrm{Geo}(0.012423\ldots,2^9-1)}\right]$ is obviously 1, since there is no other station, which can force the only one to fall asleep.

B.5 Is it worth to extend agents' memory by 1 bit?

Now, let us try to answer a question: how much should we increase the capacity of the network N in order to increase K by 1? Consider a case when capacity of the network changes from some N assumed a priori to some N^*, which is sufficient to increase the number of rounds K by 1 (i.e. when we replace N by N^* in Theorem 2.8.1). Moreover, when $\varepsilon \approx \tilde{\varepsilon}$, then it is justified to substitute $\tilde{\varepsilon}$ by ε in (2.20) in order to simplify a difference in efficiency rate for networks with capacities N^* and N (and the same ε):

$$\lg\left(\ln\left(\frac{eN^*}{\sqrt{2\tilde{\varepsilon}}}\right) - \lg\left(\ln\left(\frac{eN}{\sqrt{2\tilde{\varepsilon}}}\right)\right)\right) \approx \lg\left(\frac{\ln\left(\frac{eN^*}{\sqrt{2\tilde{\varepsilon}}}\right)}{\ln\left(\frac{eN}{\sqrt{2\tilde{\varepsilon}}}\right)}\right).$$

Assuming $\varepsilon \approx \tilde{\varepsilon}$, the right hand side of above approximation is 1 whenever $N^* = \frac{eN^2}{\sqrt{2\varepsilon}}$. Therefore, a function $B(N;\varepsilon) := \frac{eN^2}{\sqrt{2(\varepsilon - C(N,\varepsilon))}}$ may be used to provide an approximation of the capacity of the network, for which the number of rounds increases by 1 with respect to the network defined by parameters N and ε. [17] We can use the k-times composition of function $B(N;\varepsilon)$[18] to increase the number of bits by k.

Note that the closer $\tilde{\varepsilon}(N,\varepsilon)$ and $\tilde{\varepsilon}(B(N;\varepsilon),\varepsilon)$ are to ε, the more efficient the approximation become. This is the effect of reduced impact of error, which arise during the substitution ε by $\tilde{\varepsilon}$ in (2.20).

In Table B.1, we can see how K depends on the capacity of the network N and mistake frequency ε. Note that in Table B.1 the transformation $B(N,\varepsilon)$ usually increases the number of bits by 1, as expected. However, in two cases, this number remains the same. Such the situation suggests that $B(N,\varepsilon)$ slightly underestimate the capacity, which doubles L. In Table B.1 we also provided parameters $\tilde{\varepsilon}$. One can see, that when ε gets smaller, then the fraction $\frac{\tilde{\varepsilon}}{\varepsilon}$ slightly increases. Also, this ratio raises as the capacity of the network N gets higher. Both tendencies can also be seen in Table 2.1. Note also, that $-\lg\left(10^{-3}\right) \approx 10$ and $-\lg\left(10^{-6}\right) \approx 20$, so in the cases compared in Table B.1, when $N < 10^{10}$, then there are at most 4 additional bits needed with respect to the minimal threshold of $-\lceil \lg(\varepsilon) \rceil$ (compare with Section 2.5.6 and Section 2.6.4).

[17] $C(N^*,\varepsilon) < C(N,\varepsilon)$, so the affirmation of the approximation $\varepsilon \approx \tilde{\varepsilon}$ for parameter N, entails the same for capacity N^*.

[18] Here, ε is a parameter and N is an argument, so the compositions are taken with respect to the N argument.

Initial parameters				$x = \left\lceil \frac{eN^2}{\sqrt{2\tilde{\varepsilon}}} \right\rceil$		
ε	$\tilde{\varepsilon}(N,\varepsilon)$	N	K	$\tilde{\varepsilon}(x,\varepsilon)$	x	K
10^{-3}	$8.78227...\cdot 10^{-4}$	10	13	$9.99831...\cdot 10^{-4}$	6486	14
10^{-3}	$9.42318...\cdot 10^{-4}$	20	13	$9.99956...\cdot 10^{-4}$	25047	14
10^{-3}	$9.77634...\cdot 10^{-4}$	50	13	$9.99993...\cdot 10^{-4}$	153685	14
10^{-3}	$9.88930...\cdot 10^{-4}$	100	13	$9.99998...\cdot 10^{-4}$	611219	14
10^{-3}	$9.94493...\cdot 10^{-4}$	200	13	$9.99999...\cdot 10^{-4}$	2438028	14
10^{-3}	$9.97804...\cdot 10^{-4}$	500	13	$9.99999...\cdot 10^{-4}$	$1.52124...\cdot 10^{7}$	14
10^{-3}	$9.98903...\cdot 10^{-4}$	1000	13	$9.99999...\cdot 10^{-4}$	$6.08160...\cdot 10^{7}$	14
10^{-3}	$9.99452...\cdot 10^{-4}$	2000	13	$9.99999...\cdot 10^{-4}$	$2.43197...\cdot 10^{8}$	14
10^{-3}	$9.99781...\cdot 10^{-4}$	5000	13	$9.99999...\cdot 10^{-4}$	$1.51973...\cdot 10^{9}$	14
10^{-3}	$9.99890...\cdot 10^{-4}$	10000	13	$9.99999...\cdot 10^{-4}$	$6.07860...\cdot 10^{9}$	14
10^{-3}	$9.99945...\cdot 10^{-4}$	20000	13	$9.99999...\cdot 10^{-4}$	$2.43137...\cdot 10^{10}$	14
10^{-3}	$9.99978...\cdot 10^{-4}$	50000	14	$9.99999...\cdot 10^{-4}$	$1.51958...\cdot 10^{11}$	14
10^{-3}	$9.99989...\cdot 10^{-4}$	100000	14	$9.99999...\cdot 10^{-4}$	$6.07830...\cdot 10^{11}$	15
10^{-6}	$8.88574...\cdot 10^{-7}$	10	23	$9.99995...\cdot 10^{-7}$	203907	24
10^{-6}	$9.47219...\cdot 10^{-7}$	20	23	$9.99999...\cdot 10^{-7}$	789977	24
10^{-6}	$9.79534...\cdot 10^{-7}$	50	23	$9.99999...\cdot 10^{-7}$	4.855230	24
10^{-6}	$9.89870...\cdot 10^{-7}$	100	23	$9.99999...\cdot 10^{-7}$	$1.93193...\cdot 10^{7}$	24
10^{-6}	$9.94961...\cdot 10^{-7}$	200	23	$9.99999...\cdot 10^{-7}$	$7.70791...\cdot 10^{7}$	24
10^{-6}	$9.97990...\cdot 10^{-7}$	500	23	$9.99999...\cdot 10^{-7}$	$4.81012...\cdot 10^{8}$	24
10^{-6}	$9.98996...\cdot 10^{-7}$	1000	23	$9.99999...\cdot 10^{-7}$	$1.92308...\cdot 10^{9}$	24
10^{-6}	$9.99498...\cdot 10^{-7}$	2000	24	$9.99999...\cdot 10^{-7}$	$7.69039...\cdot 10^{9}$	24
10^{-6}	$9.99799...\cdot 10^{-7}$	5000	24	$9.99999...\cdot 10^{-7}$	$4.80577...\cdot 10^{10}$	25
10^{-6}	$9.99900...\cdot 10^{-7}$	10000	24	$9.99999...\cdot 10^{-7}$	$1.92221...\cdot 10^{11}$	25
10^{-6}	$9.99950...\cdot 10^{-7}$	20000	24	$9.99999...\cdot 10^{-7}$	$7.68865...\cdot 10^{11}$	25
10^{-6}	$9.99980...\cdot 10^{-7}$	50000	24	$9.99999...\cdot 10^{-7}$	$4.80534...\cdot 10^{12}$	25
10^{-6}	$9.99990...\cdot 10^{-7}$	100000	24	$9.99999...\cdot 10^{-7}$	$1.92213...\cdot 10^{13}$	25

Table B.1: A comparison of the number of bits K with respect to different capacities of a network, obtained according to Theorem 2.8.1 with $\varepsilon \in \{10^{-3}, 10^{-6}\}$, together with appropriate $\tilde{\varepsilon}$ parameters.

Appendix C

Limitation of specific sums

We this appendix, we are going to find upper bounds of some sums.

Lemma 21. *Let* $p \in \left(0, \frac{1}{2}\right)$. *Then*

$$\sum_{a=2}^{n}(a+1)^2 p^a < 9\frac{p^2}{1-p} + \frac{p^2\left(3+2p-3p^2\right)}{(1-p)^3} \ .$$

Proof. Denote the left hand side by υ. Then

$$\begin{aligned}\upsilon &= \sum_{a=3}^{n+1} a\frac{\partial p^a}{\partial p} = \frac{\partial}{\partial p}\left(p\sum_{a=3}^{n+1} a p^{a-1}\right) \\ &= \frac{\partial}{\partial p}\left(p\frac{\partial}{\partial p}\left(\sum_{a=3}^{n+1} p^a\right)\right) = \frac{\partial}{\partial p}\left(p^3\frac{1-p^{n-1}}{1-p}\right) + p\frac{\partial^2}{\partial p^2}\left(p^3\frac{1-p^{n-1}}{1-p}\right) \ .\end{aligned}$$

Now let us calculate the first derivative:

$$\begin{aligned}\frac{\partial}{\partial p}\left(p^3\frac{1-p^{n-1}}{1-p}\right) &= 3p^2\frac{1-p^{n-1}}{1-p} + p^3\frac{1-p^{n-1}-(n-1)p^{n-2}(1-p)}{(1-p)^2} \\ &= 3p^2\frac{1-p^{n-1}}{1-p} + p^3\frac{1-p^{n-2}((n-1)-(n-2)p)}{(1-p)^2} \ .\end{aligned}$$

Then, the second one:

$$\begin{aligned}\frac{\partial^2}{\partial p^2}\left(p^3\frac{1-p^{n-1}}{1-p}\right) &= \frac{\partial}{\partial p}\left(3p^2\frac{1-p^{n-1}}{1-p} + p^3\frac{1-p^{n-2}((n-1)-(n-2)p)}{(1-p)^2}\right) \\ &= 6p\frac{1-p^{n-1}}{1-p} + 3p^2\frac{1-p^{n-2}((n-1)-(n-2)p)}{(1-p)^2} \\ &+ 3p^2\frac{1-p^{n-2}((n-1)-(n-2)p)}{(1-p)^2} \\ &+ \frac{p^3}{(1-p)^4}\left(-p^{n-3}(n-2)((n-1)-(n-2)p)(1-p)^2\right. \\ &\left.+2(1-p)\left(1-p^{n-2}((n-1)-(n-2)p)\right)\right) = \dots\end{aligned}$$

$$\begin{aligned}\dots &= 6p\frac{1-p^{n-1}}{1-p} + 6p^2\frac{1-p^{n-2}((n-1)-(n-2)p)}{(1-p)^2} \\ &- \frac{p^3}{(1-p)^2}p^{n-3}(n-2)((n-1)-(n-2)p) \\ &+ \frac{p^3}{(1-p)^3}2\left(1-p^{n-2}((n-1)-(n-2)p)\right) \ .\end{aligned}$$

Now, one can easily calculate υ, however, let us realize that $(n-2)p < \frac{n-2}{2} < n-1$, so

$$\begin{aligned}\upsilon &< \frac{3p^2}{1-p} + \frac{p^3}{(1-p)^2} + p\left(\frac{6p}{1-p} + \frac{6p^2}{(1-p)^2} + \frac{2p^3}{(1-p)^3}\right) \\ &= \frac{9p^2}{1-p} + \frac{7p^3}{(1-p)^2} + \frac{2p^4}{(1-p)^3} = \frac{9p^2-11p^3+4p^4}{(1-p)^3} \ .\end{aligned}$$

□

Lemma 22. *Let $p \in \left(0, \frac{1}{2}\right)$. Then*

$$\sum_{a=2}^{n}\left(p^{a-1} + \frac{(a+1)^2p^a\ln 2}{12}\right) < \frac{p}{(1-p)^2} \ .$$

Proof. Denote the sum by ξ. Then From Lemma 21 we get

$$\begin{aligned}\xi &= \left(p\frac{1-p^{n-1}}{1-p} + \ln 2\frac{9p^2-11p^3+4p^4}{12(1-p)^3}\right) \\ &< \frac{p+p^2(\frac{3\ln 2}{4}-2)+p^3(1-\frac{11\ln 2}{12})+p^4\frac{\ln 2}{3}}{(1-p)^3} \\ &= \frac{p-p^2+p^2\left[\frac{3\ln 2}{4}-1+p(1-\frac{11\ln 2}{12})+p^2\frac{\ln 2}{3}\right]}{(1-p)^3} \\ &< \frac{p-p^2}{(1-p)^3} = \frac{p}{(1-p)^2} \ ,\end{aligned}$$

because $g(x) := \frac{3\ln 2}{4} - 1 + x(1-\frac{11\ln 2}{12}) + x^2\frac{\ln 2}{3}$ is ascending in interval $(0, 0.5)$.[1] and $g(0.5) = \frac{3\ln 2}{8} - \frac{1}{2} < 0$. □

[1] Since $1 - \frac{11\ln 2}{12} > 0$ and $\frac{\ln 2}{3} > 0$.

Appendix D

Special cases of Power Law of Update

This appendix is devoted to rectifications of results from Section 3.3 for linear and sublinear cases.

D.1 Specific linear case

We begin this section with the solution of some recurrence relation for $\alpha = 1$:

Lemma 23. *Suppose that* $0 < g \leqslant 2$, $\alpha_n = \min\left(1, \frac{g}{n}\right)$, $\mathbb{E}[K_0] = 0$ *and* $\mathbb{E}[K_{n+1}] = 1 + (1 - \frac{g}{n+1})\mathbb{E}[K_n]$ *for each* $n \in \mathbb{N}_0$. *Then*

$$\mathbb{E}[K_n] = \frac{n+1}{g+1} + (-1)^{n+1}\frac{\binom{g-1}{n}}{g+1} \,. \tag{D.1}$$

Let us note that originally the above formula was obtained via technique from next Section D.2. This experience let us find slightly shorter proof, which we derived thanks to a method utilizing Z-transform.[1] However, the most concise proof can be done by an easy induction:

Proof. For our convenience, we denote $x_n := \mathbb{E}[K_n]$. Note that $x_0 = 0$. Moreover, when we assume that Eq. (D.1) is fulfilled for some $n \in \mathbb{N}_0$, then by definitions from Section 1.2.8 we get:

$$\begin{aligned}
1 + \left(1 - \frac{g}{n+1}\right) x_n &= 1 + \frac{n+1}{g+1} - (-1)^n \frac{\binom{g-1}{n}}{g+1} - \frac{g}{g+1} - (-1)^{n+1}\frac{\binom{g-1}{n}}{g+1}\frac{g}{n+1} \\
&= \frac{n+2}{g+1} - (-1)^{n+1}\frac{\binom{g-1}{n}}{g+1}\left(\frac{g}{n+1} - \frac{n+1}{n+1}\right) \\
&= \frac{n+2}{g+1} - (-1)^{n+1}\frac{\binom{g-1}{n+1}}{g+1} \,.
\end{aligned}$$

[1] For a definition and useful techniques see [67].

Therefore $\mathbb{E}[K_{n+1}]$ satisfies Eq. (D.1) as well, what ends the proof. □

Nevertheless, for a completeness, we also attach the modified version of Z-transform method.[2]

Proof. Note that when $g \leqslant 2$, then $n \geqslant g-1$ implies $n \geqslant 1$, hence the foregoing formula: $x_{n+1} = 1 + (1 - \frac{g}{n+1})x_n$ is satisfied for all $n \in \mathbb{N}$.

Realize that then we can rewrite the previous relation to

$$(n+1)x_{n+1} = (n+1) + nx_n + (1-g)x_n \ . \tag{D.2}$$

Let us consider the function $f(z) = \sum_{n=0}^{\infty} x_n z^n$,[3] which is modified Z-transform of $(x_n)_n$. Notice that $1 \leqslant \mathbb{E}[K_n] = x_n \leqslant n$, hence the modified Z-transform is convergent for $|z| < 1$.

From Eq. (D.2) with $x_0 = 0$ we can get:

$$\sum_{n=0}^{\infty}(n+1)x_{n+1}z^n = \sum_{n=0}^{\infty}(n+1)z^n + \sum_{n=0}^{\infty}nx_nz^n + (1-g)\sum_{n=0}^{\infty}x_nz^n \ .$$

Since $(n+1)z^n = \frac{\partial}{\partial z}z^{n+1}$ and $nz^n = z\frac{\partial}{\partial z}z^n$, we obtain:

$$\frac{\partial}{\partial z}(f(z) - x_0) = \frac{\partial}{\partial z}\frac{z}{1-z} + z\frac{\partial}{\partial z}f(z) + (1-g)f(z) \ .$$

Note that $\frac{\partial}{\partial z}\frac{z}{1-z} = \frac{1}{(1-z)^2}$. We transmit all the expressions with $\frac{\partial}{\partial z}f(z)$ and $f(z)$ on the left side and all the others on the right hand size of the equation and further we multiply both sides by $(1-z)^{-g}$:

$$(1-z)^{1-g}\frac{\partial}{\partial z}f(z) - \frac{1-g}{1-z}(1-z)^{1-g}f(z) = \frac{1}{(1-z)^{g+2}} \ .$$

Realize that $\frac{\partial}{\partial z}(1-z)^{1-g} = -\frac{1-g}{1-z}(1-z)^{1-g}$, therefore

$$\frac{\partial}{\partial z}((1-z)^{1-g}f(z) = \frac{1}{(1-z)^{g+2}} \ .$$

We integrate both sides of the above to obtain

$$f(z) = (1-z)^{g-1}\int_0^z \frac{1}{(1-t)^{g+2}}dt = (1-z)^{g-1}\left(\frac{(1-z)^{-g-1}}{g+1} - \frac{1}{g+1}\right)$$
$$= \frac{(1-z)^{-2}}{(g+1)} - (1-z)^{g-1} \ .$$

[2] Namely, we declare a series with terms z^n instead of z^{-n}.

[3] We assume that $x_0 = 0$ in order to extend the applicability of Eq. (D.2) to $n \in \mathbb{N}_0$.

Note, that we have already showed that $(1-z)^{-2} = \sum_{n=0}^{\infty}(n+1)z^n$, so from Fact 1.2.6 we finally attain

$$[z^n]f(z) = x_n = \frac{n+1}{1+g} - (-1)^n \frac{\binom{g-1}{n}}{g+1} \ .$$

□

Theorem D.1.1. *Suppose that* $0 < g \leqslant 2$ *or* $g \in \mathbb{N}$ *and* $\alpha_n = \min\left(1, \frac{g}{n+1}\right)$. *Then*

$$\mathbb{E}[K_n] = \frac{1}{1+g}\left((n+1) - n^{-g}\frac{\Gamma(g)\sin(g\pi)}{\pi} + \mathrm{O}\left(n^{-1-g}\right) \ ,\right)$$

as $n \to \infty$.

Proof. Suppose first that $0 < g \leqslant 2$. As we mentioned in the proof of Lemma 23, the assumption $0 < g \leqslant 2$ entails $\alpha_n = \frac{g}{n+1}$ for each $n \in \mathbb{N}$. Therefore we may use Lemma 23 directly. The following follows from Fact 1.2.7

$$\binom{g-1}{n} = (-1)^{n+1}\frac{\Gamma(g)\sin(\pi(n-g+1))}{\pi n^g} + \mathrm{O}\left(n^{-g-1}\right) \ ,$$

as $n \to \infty$.

Realize that $\sin(\pi(n-g+1)) = \sin(\pi(n+1))\cos(\pi g) - \sin(\pi g)\cos(\pi(n+1)) = (-1)^{n+2}\sin(\pi g)$, so we finally get

$$\mathbb{E}[K_n] = \frac{n+1}{1+g} - n^{-g}\frac{\Gamma(g)\sin(g\pi)}{(1+g)\pi} + \mathrm{O}\left(n^{-1-g}\right) \ ,$$

as $n \to \infty$. □

If the parameter g is greater that 2, then for $n \leqslant g-1$ we have $\alpha_n = 1$, hence if $n \leqslant \lfloor g-1 \rfloor$ then we have $K_n = 1$. Therefore the recursion $\mathbb{E}[K_{n+1}] = 1 + (1 - \frac{g}{n+1})\mathbb{E}[K_n]$ starts working from $n = \lfloor g \rfloor$. Note that one can utilize Z-transform (or any similar method) in order to obtain results analogous to Lemma 23 and Theorem D.1.1, for $g > 2$. For $g \in \mathbb{N}\backslash\{1,2\}$, the result is quite intuitive: $\mathbb{E}[K_n] = 1$ for $n \in [g]$ and $\mathbb{E}[K_n] = \frac{n+1}{g+1}$ otherwise (note that the second part coincides with the formula (D.1) in this case as well). However (D.1) is not correct for $g \in [2,\infty)\backslash\mathbb{N}$ and the appropriate Z-transform method leads to quite challenging formulas with incomplete Beta function.[4]

D.2 Heuristic recursive approach with formal series

[4] See [1] for definition and properties.

D.2.1 Problem for a case $\alpha = \frac{1}{2}$

Let us denote $x_n := \mathbb{E}[K_n]$ for convenience. Then $x_1 = 1$ and let us assume $\alpha_n = \min\left(1, \frac{g}{\sqrt{n}}\right)$, hence according to Theorem 3.3.1, we get

$$x_{n+1} = 1 + \left(1 - \frac{g}{\sqrt{n+1}}\right) x_n \tag{D.3}$$

for $n \geqslant \lceil g^2 \rceil - 1$. As we have seen in Section 3.3.5, $x_n \sim \frac{\sqrt{n}}{g}$, as $n \to \infty$. In this section we are going to find much better asymptotics.

Solution

At first, realize that there is no Laurent series (see Section 1.2.4 for definition), which satisfies (D.3), as $n \to \infty$. Indeed, if there is one, then its application to (D.3) results in an appearance of monomials of the form $c_k n^{k+\frac{1}{2}}$. However, then we can obtain that all the coefficients of Laurent series are zeros, conversely to $x_1 = 1$.

Therefore let us postulate the foregoing form of a solution:

$$x_n = \sum_{k=0}^{\infty} a_k n^{\frac{1}{2}-\frac{k}{2}} \ . \tag{D.4}$$

We evaluate (D.3) with (D.4). Let us start with simplifying both sides:

$$\begin{aligned} x_{n+1} = \sum_{k=0}^{\infty} a_k (n+1)^{\frac{1}{2}-\frac{k}{2}} &= \sum_{k=0}^{\infty} a_k n^{\frac{1}{2}-\frac{k}{2}} \left(1 + \frac{1}{n}\right)^{\frac{1}{2}-\frac{k}{2}} \\ &\overset{\text{Fact1.2.6}}{=} \sum_{k=0}^{\infty} a_k n^{\frac{1}{2}-\frac{k}{2}} \sum_{s=0}^{\infty} \binom{\frac{1}{2}-\frac{k}{2}}{s} n^{-s} \end{aligned}$$

and

$$\begin{aligned} 1 + \left(1 - \frac{g}{\sqrt{n+1}}\right) x_n &= 1 + \sum_{k=0}^{\infty} a_k n^{\frac{1}{2}-\frac{k}{2}} - \sum_{k=0}^{\infty} g a_k n^{-\frac{k}{2}} \left(1 + \frac{1}{n}\right)^{-\frac{1}{2}} \\ &\overset{\text{Fact1.2.6}}{=} 1 + \sum_{k=0}^{\infty} a_k n^{\frac{1}{2}-\frac{k}{2}} - \sum_{k=0}^{\infty} g a_k n^{-\frac{k}{2}} \sum_{s=0}^{\infty} \binom{-\frac{1}{2}}{s} n^{-s} \ . \end{aligned}$$

A comparison of both formulas provides:

$$\sum_{k=0}^{\infty}\sum_{s=0}^{\infty} \binom{\frac{1}{2}-\frac{k}{2}}{s} a_k n^{\frac{1}{2}-\frac{k}{2}-s} = 1 + \sum_{k=0}^{\infty} a_k n^{\frac{1}{2}-\frac{k}{2}} - \sum_{k=0}^{\infty}\sum_{s=0}^{\infty} \binom{-\frac{1}{2}}{s} g a_k n^{-\frac{k}{2}-s} \tag{D.5}$$

Let us consider the above formulas formally. Thence we are capable to swap the series when needed. Note that, when we extract an element of the second series

corresponding to $s = 0$ on the left hand side of Eq. (D.5), we obtain exactly one of the series from the right hand side of this formula, so we simply reduce them. We divide Eq. (D.5) in a system of two equations – one with integer powers of n – and the second with non-integer ones. This way we receive the foregoing expressions:

$$\begin{cases} \displaystyle\sum_{k=0}^{\infty}\sum_{s=1}^{\infty}\binom{\frac{1}{2}-k}{s}a_{2k}n^{\frac{1}{2}-k-s} &= -\displaystyle\sum_{k=0}^{\infty}\sum_{s=0}^{\infty}\binom{-\frac{1}{2}}{s}ga_{2k+1}n^{-k-\frac{1}{2}-s}\ , \\ \displaystyle\sum_{k=0}^{\infty}\sum_{s=1}^{\infty}\binom{-k}{s}a_{2k+1}n^{-k-s} &= 1-\displaystyle\sum_{k=0}^{\infty}\sum_{s=0}^{\infty}\binom{-\frac{1}{2}}{s}ga_{2k}n^{-k-s}\ . \end{cases}$$

Now, we rearrange the both formulas to the following form:

$$\sum_{k=0}^{\infty} ga_{2k+1}n^{-k-\frac{1}{2}} + \sum_{k=0}^{\infty}\left(\frac{1}{2}-k\right)a_{2k}n^{-k-\frac{1}{2}} = -\sum_{k=0}^{\infty}\sum_{s=1}^{\infty}\left[\binom{-\frac{1}{2}}{s}ga_{2k+1}n^{-k-\frac{1}{2}-s} + \binom{\frac{1}{2}-k}{s+1}a_{2k}n^{-\frac{1}{2}-k-s}\right],$$

$$\sum_{k=0}^{\infty} ga_{2k}n^{-k} = 1-\sum_{k=0}^{\infty}\sum_{s=1}^{\infty}\left[\binom{-k}{s}a_{2k+1}n^{-k-s} + \binom{-\frac{1}{2}}{s}ga_{2k}n^{-k-s}\right].$$

Now, for each $r \in \mathbb{N}_0$ we extract terms with $n^{-r-\frac{1}{2}}$ from the upper equation and n^{-r} from the latter one and obtain a system:

$$\begin{bmatrix} g & 0 \\ \frac{1}{2}-r & g \end{bmatrix}\begin{bmatrix} a_{2r} \\ a_{2r+1} \end{bmatrix} = \begin{bmatrix} \mathbf{1}_{\{0\}}(r) - \displaystyle\sum_{s=1}^{r}\left[\binom{-r+s}{s}a_{2r-2s+1} + \binom{-\frac{1}{2}}{s}ga_{2r-2s}\right] \\ -\displaystyle\sum_{s=1}^{r}\left[\binom{-\frac{1}{2}}{s}ga_{2r-2s+1} + \binom{\frac{1}{2}-r+s}{s+1}a_{2r-2s}\right] \end{bmatrix}.$$

The most left matrix of the above equation is triangular, hence obviously invertible. Note that on the right hand side of the above equation, the coefficients a_i appears only for $i < 2r$. Therefore, the above formula provide a recursive step. When $r = 0$, we attain simply

$$\begin{bmatrix} g & 0 \\ \frac{1}{2} & g \end{bmatrix}\begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}\ .$$

Thence $a_0 = \frac{1}{g}$ and $a_1 = -\frac{1}{2g}a_0 = -\frac{1}{2g^2}$. Also this shows that the recurrence is well-defined, what implies that (D.4) is appropriate. We can easily find the first coefficients and, for instance, obtain the beneath asymptotics

$$x_n = \mathbb{E}[K_n] = \frac{\sqrt{n}}{g} - \frac{1}{2g^2} + \frac{1}{2g\sqrt{n}} - \frac{1}{8g^2n} + \mathrm{O}\left(n^{-\frac{3}{2}}\right)\ .$$

D.2.2 Problem for cases $\alpha = \frac{r}{q}$

Let us extend the solution from Section D.2.1 to a case, when $\alpha \in (0,1) \cap \mathbb{Q}$. Using the same notation as in Section D.2.1, we have $x_1 = 1$ and

$$x_{n+1} = 1 + \left(1 - g(n+1)^{-\frac{r}{q}}\right) x_n \ , \tag{D.6}$$

where $n \geqslant \left\lceil g^{\frac{q}{r}} \right\rceil - 1$, $q \in \mathbb{N}$, $r \in [q-1]$ and $\mathrm{GCD}(r,q) = 1$ is satisfied. [5] In Section 3.3.5 we have showed that $x_n \sim n^{\frac{r}{q}}$ and now we are going to find much better asymptotics.

Solution

Analogously to the solution from Section D.2.1, we postulate the foregoing form of a solution:

$$x_n = \sum_{k=0}^{\infty} a_k n^{\frac{r}{q} - \frac{k}{q}} \ . \tag{D.7}$$

We evaluate (D.6) with (D.7). Simplifications and erasing the same components led to an equation similar to (D.5):

$$\sum_{k=0}^{\infty}\sum_{s=1}^{\infty} \binom{\frac{r}{q} - \frac{k}{q}}{s} a_k n^{\frac{r}{q} - \frac{k}{q} - s} = 1 - \sum_{k=0}^{\infty}\sum_{s=0}^{\infty} \binom{-\frac{r}{q}}{s} g a_k n^{-\frac{k}{q} - s} \ . \tag{D.8}$$

Let $\left(\left[n^{\mathbb{N} - \frac{p}{q}}\right](.)\right)_{p \in [0:q-1]}$ be the sequence of n-term extractors (see Section 1.2.1 for definition).[6] We divide (D.8) into q equations – every corresponded to one of above extractors – what results in receiving the foregoing system of q equations:

$$\sum_{k=0}^{\infty}\sum_{s=1}^{\infty} \binom{-k}{s} a_{qk+r} n^{-k-s} = 1 - \sum_{k=0}^{\infty}\sum_{s=0}^{\infty} \binom{-\frac{r}{q}}{s} g a_{qk} n^{-k-s} \ ,$$

$$\sum_{k=0}^{\infty}\sum_{s=1}^{\infty} \binom{-k - \frac{p}{q}}{s} a_{qk+r+p} n^{-k-s-\frac{p}{q}} = - \sum_{k=0}^{\infty}\sum_{s=0}^{\infty} \binom{-\frac{r}{q}}{s} g a_{qk+p} n^{-k-s-\frac{p}{q}},$$

$$\sum_{k=0}^{\infty}\sum_{s=1}^{\infty} \binom{\frac{r}{q} - k - \frac{p'}{q}}{s} a_{qk+p'} n^{\frac{r}{q} - k - \frac{p'}{q} - s}$$
$$= - \sum_{k=0}^{\infty}\sum_{s=0}^{\infty} \binom{-\frac{r}{q}}{s} g a_{q(k+1)-r+p'} n^{-k-s-\frac{q-r}{q}-\frac{p'}{q}} \ ,$$

where $p \in [q-r-1]$ and $p' \in [0 : r-1]$.

[5] See Section 1.2.1.

[6] Roughly speaking, each of the mentioned extractors acts like $\left[n^{-\frac{p}{q} \bmod 1}\right]$.

We rearrange all the equations to the following forms:

$$\sum_{k=0}^{\infty} g a_{qk} n^{-k} = 1 - \sum_{k=0}^{\infty}\sum_{s=1}^{\infty}\left[\binom{-\frac{r}{q}}{s} g a_{qk} + \binom{-k}{s} a_{qk+r}\right] n^{-k-s} ,$$

$$\sum_{k=0}^{\infty} g a_{qk+p} n^{-k-\frac{p}{q}} = -\sum_{k=0}^{\infty}\sum_{s=1}^{\infty}\left[\binom{-\frac{r}{q}}{s} g a_{qk+p} + \binom{-k-\frac{p}{q}}{s} a_{qk+r+p}\right] n^{-k-s-\frac{p}{q}} ,$$

$$\begin{aligned}&\sum_{k=0}^{\infty}\left[g a_{q(k+1)-r+p'} + \frac{r-p'-qk}{q} a_{qk+p'}\right] n^{-k-\frac{q-r}{q}-\frac{p'}{q}} \\ &\quad = -\sum_{k=0}^{\infty}\sum_{s=1}^{\infty}\left[\binom{-\frac{r}{q}}{s} g a_{q(k+1)-r+p'} + \binom{\frac{r}{q}-k-\frac{p'}{q}}{s+1} a_{qk+p'}\right] n^{-k-s-\frac{q-r+p'}{q}} ,\end{aligned}$$

where $p \in [q-r-1]$ and $p' \in [0:r-1]$.

Short remark on matrix notation Let $\mathbb{O}_i$ denotes a square matrix of dimension $i \times i$ consisting of zeros and $\mathbb{O}_{i\times j}$ be a matrix of dimension $i \times j$ full of zeros. Moreover $\mathbb{I}_i$ is a diagonal matrix of dimension $i \times i$, which has only ones on diagonal. Let $\operatorname{diag}(v)$ be a diagonal matrix with coordinates of a vector v on the diagonal. When $a \leqslant b$, then $[v_i]_{i=a}^{b}$ denotes a vertical vector ($1 \times (b-a+1)$-matrix) with coordinates $v_a, v_{a+1}, \ldots, v_b$.

Let us define $B_{r,q}$ as the block matrix:

$$\begin{bmatrix} \mathbb{O}_{r\times(q-r)} & \mathbb{O}_{q-r} \\ \operatorname{diag}\left(\frac{r}{q}-k, \frac{r-1}{q}-k, \ldots, \frac{1}{q}-k\right) & \mathbb{O}_{(q-r)\times r} \end{bmatrix} ,$$

$\bar{a}$ as the vertical vector (in block matrix notation):

$$\begin{bmatrix} [a_{qk}] \\ [a_{qk+p}]_{p=1}^{q-r-1} \\ \left[a_{q(k+1)-r+p'}\right]_{p'=0}^{r-1} \end{bmatrix}$$

and $A_{r,q} := g\mathbb{I}_q + B_{r,q}$. We extract terms corresponding to n^{-k}, $n^{-k-\frac{p}{q}}$ or $n^{-k-\frac{p'}{q}}$ from every of q equations and get:

$$A_{r,q}\bar{a} = \begin{bmatrix} \mathbf{1}_{\{0\}}(k) - \sum_{s=1}^{k}\left[\binom{-k+s}{s} a_{qk-qs+r} + \binom{-\frac{r}{q}}{s} g a_{qk-qs}\right] \\ \left[-\sum_{s=1}^{k}\left[\binom{-k+s-\frac{p}{q}}{s} a_{qk-qs+p+r} + \binom{-\frac{r}{q}}{s} g a_{qk-qs+p}\right]\right]_{p=1}^{q-r-1} \\ \left[-\sum_{s=1}^{k}\left[\binom{\frac{r}{q}-k+s-\frac{p'}{q}}{s+1} a_{qk-qs+p'} + \binom{-\frac{r}{q}}{s} g a_{qk-qs-r+p'}\right]\right]_{p'=0}^{r-1} \end{bmatrix} ,$$

where $k \in \mathbb{N}_0$. $A_{r,q}$ is the triangular matrix, hence invertible, for all r, q, so the recurrence is well-defined, what entails that (D.7) is an appropriate from of the solution of (D.6) with coefficients defined by above system of equations. We can easily find the first coefficients:

a_0	a_p for $p \in [q-r-1]$	a_{q-r}	$a_{q-r+p'}$ for $p' \in [r-1]$	a_q
$\frac{1}{g}$	0	$-\frac{r}{qg^2}$	$-\frac{r-p'}{qg} \cdot a_{p'}$	$-\frac{r}{qg}$

That gives the beneath asymptotics

$$x_n = \mathbb{E}[K_n] = \frac{1}{g} n^{\frac{r}{q}} - \frac{r}{qg^2} n^{\frac{2r}{q}-1} + \mathrm{O}\left(n^{\max\left(\frac{3r}{q}-2, \frac{r}{q}-1\right)}\right) .$$

For instance, when $r = 1$ and $q = 5$, one may obtain

$$\mathbb{E}[K_n] = \frac{1}{g} n^{\frac{1}{5}} - \frac{1}{5g^2} n^{-\frac{3}{5}} - \frac{1}{5g} n^{-\frac{4}{5}} - \frac{3}{25g^2} n^{-\frac{7}{5}} + \mathrm{O}\left(n^{-\frac{8}{5}}\right)$$

and when $r = 2$ and $q = 3$, we can get:

$$\mathbb{E}[K_n] = \frac{1}{g} n^{\frac{2}{3}} - \frac{2}{3g^2} n^{\frac{1}{3}} + \frac{1}{9g^3} - \frac{1}{3g} n^{-\frac{1}{3}} + \frac{1}{9g^2} \left(\frac{1}{3g^2} - 2\right) n^{-\frac{2}{3}} + \mathrm{O}\left(n^{-1}\right) .$$

Remark Note that the method presented in Section D.2.2 can be extended to a case $r = q = 1$, what provides a recursive relation:

$$(g + 1 - k) a_k = \mathbf{1}_{\{0\}}(k) - \sum_{s=1}^{k} \left[\binom{1-k+s}{s} + \binom{-1}{s} g\right] a_{k-s}$$

for $k \geqslant g - 1$ and $\mathbb{E}[K_n]$ of the form $\sum_{k=0}^{\infty} a_k n^{1-k}$, where $\alpha_n = \min\left(1, \frac{g}{n}\right)$.

Appendix E

Technical Results for Morris Counter

For the sake of completeness, we present here proofs of all gory lemmas that are not directly connected to Theorem 4.5.1. For convenience, some of computations are supported by Wolfram Mathematica ver.11.3 ([60]). Whenever we obtain a result in this manner, we indicate it by $\overset{W}{=}$ sign. Usually results are precise, however in some cases, final forms are attained numerically.
Let us commence with several definitions used throughout this appendix. An increasing sequence $\prod_{i=1}^{k}\left(1-2^{-i}\right)^{-1}$ that arised in Theorem 4.5.2 will be indicated by r_k and we denote its limit $\prod_{i=1}^{\infty}\left(1-2^{-i}\right)^{-1} = 3.46274\ldots$ by R. We often struggle with expressions of a pattern $1-\frac{1}{y}$, so we denote this function as $a(y)$ to simplify formulas.

E.1 Proofs of δ-lemmas

E.1.1 The proof of Lemma 12

Proof. At first, we want to bound a lower tail of distribution of M_n, namely $\Pr[M_n \leqslant \lceil \lg(n) \rceil - 5]$. Here we would like to find a sufficient upper limitation for the above probability. Assume that $l \leqslant \lceil \lg(n) \rceil - 5$. Consider the probability that M_n has value l:

$$p_{n,l} \overset{\text{Thm } 4.5.2}{\leqslant} \sum_{j=0}^{l-1} 2^{-\frac{j(j-1)}{2}} \left(1-2^{-(l-j)}\right)^n r_j r_{l-1-j} \leqslant R^2\left(1-2^{-l}\right)^n \sum_{j=0}^{l-1} \sqrt{2}^{-j+1}$$
$$\leqslant R^2 \frac{2}{\sqrt{2}-1} \exp\left(-n2^{-l}\right) = R^2(2\sqrt{2}+2)\exp\left(-n2^{-l}\right).$$

Remark that the above restraint is useless when $l \geqslant \lg(n) - 2$, so it cannot be employed to obtain a reasonable bound for an symmetrical upper tail. However, the aforementioned formula will help us to limit the left tail of the distribution of M_n:

$$\begin{aligned}\delta_1 = \sum_{l=1}^{\lceil \lg(n) \rceil - 5} \Pr[M_n = l] &\leqslant R^2(2\sqrt{2}+2) \sum_{l=1}^{\lceil \lg(n) \rceil - 5} \exp\left(-n2^{-l}\right) \\ &\leqslant R^2(2\sqrt{2}+2) \sum_{k=4}^{\infty} \exp\left(-2^k\right) \leqslant R^2(2\sqrt{2}+2) \sum_{k=1}^{\infty} e^{-16k} \\ &= R^2(2\sqrt{2}+2) \frac{e^{-16}}{1-e^{-16}} = 0.000006515315\ldots \ .\end{aligned}$$

□

E.1.2 The proof of Lemma 13

Proof. Actual goal is to limit the upper tail, that is $\Pr[M_n \geqslant \lceil \lg(n) \rceil + 5]$. Consider a process $X = (X_k, k \in [0 : n])$. Let X initially follow the incrementation rule $\Pr[X_k = k+1] = 1$ for $k \in [0 : \lceil \lg n \rceil + 1]$. Afterwards, let this Markov chain imitate the transition rule of Morris Counter, that is

$$\Pr[X_{k+1} = m+1 | X_k = m] = \frac{1}{2^m} = 1 - \Pr[X_{k+1} = m | X_k = m]$$

for $k \geqslant \lceil \lg(n) \rceil + 1$. Naturally, for $k \leqslant \lceil \lg(n) \rceil + 1$, we have $X_k \geqslant M_k$, so we may couple realizations of these two processes in such a way that whenever X is incremented, then so is M and if M does not change, then X does not rise as well.[1]

To abbreviate the expressions let us denote $m = n - \lceil \lg(n) \rceil - 1$ and

$$\mu_c = \Pr[X_{k+1} = \lceil \lg(n) \rceil + c + 1 | X_k = \lceil \lg(n) \rceil + c] = \frac{1}{2^{\lceil \lg(n) \rceil + c}} = 1 - \nu_c \ ,$$

for any $c \in \mathbb{Z}$. Moreover, let us consider a three-dimensional discrete simplex ${\mathrm{Sim}_k}^{(3)}$ (see Section 1.2.4 for definition):

The coupling encountered above, ensures us that

$$\begin{aligned}\delta_2 \leqslant \Pr[X_n \geqslant \lceil \lg(n) \rceil + 5] &= \sum_{\bar{l} \in {\mathrm{Sim}_{m-3}}^{(3)}} {\nu_2}^{l_1} \mu_2 {\nu_3}^{l_2} \mu_3 {\nu_4}^{l_3} \mu_4 \leqslant \sum_{\bar{l} \in {\mathrm{Sim}_{m-3}}^{(3)}} \frac{1}{2^{3\lceil \lg(n) \rceil + 9}} \\ &= \sum_{k=0}^{m-3} \binom{k+3}{2} \frac{1}{2^{3\lceil \lg(n) \rceil + 9}} \leqslant \frac{1}{2^{10} n^3} \sum_{k=3}^{m} k^2 - k \ .\end{aligned}$$

[1] Note that X has at most the same probability of a positive incrementation as M at any point of time.

Realize that $\sum_{k=3}^{m} k = \frac{(m-2)(m+3)}{2}$ and $\sum_{k=3}^{m} k^2 = \frac{(m-2)(2m^2+7m+15)}{6}$,[2] so

$$\begin{aligned}\delta_2 &\leqslant \frac{1}{2^{10}n^3}\frac{1}{6}(m-2)(2m^2+4m+6) = \frac{1}{3\cdot 2^{10}n^3}(m^3-m-6)\\ &\leqslant \frac{m^3}{3\cdot 2^{10}n^3} \leqslant \frac{1}{3\cdot 2^{10}} = 0.000325521\ldots\ .\end{aligned}$$

Note that when $m < 3$ (that is, when $n < 7$), then the above sums are empty, but on the other hand $\lceil \lg(n) \rceil + 5 > n+1$, so the inequality is trivially true. □

E.2 Main lemma

Auxiliary lemmas

Next two lemmas are useful in a proof of main Lemma 26:

Lemma 24. *Let $c > \frac{1}{y} > 0$ and $x \in \mathbb{N}$. Then*

$$a(2cy)^{2x} \geqslant a(cy)^{x-1}\left(a(cy) + \frac{x}{4c^2y^2}\right)$$

and

$$a(cy)^{x} \geqslant a(2cy)^{2x-2}\left(a(2cy)^2 - \frac{x}{4c^2y^2}\right)\ .$$

Proof.

$$a(2cy)^{2x} - a(cy)^x = \left(1 - \frac{1}{cy} + \frac{1}{4c^2y^2} - 1 + \frac{1}{cy}\right)\sum_{i=0}^{x-1} a(2cy)^{2i}a(cy)^{x-i-1}\ .$$

Hence we obtained two inequalities: $a(2cy)^{2x} - a(cy)^x \geqslant \frac{x}{4c^2y^2}a(cy)^{x-1}$ and $a(2cy)^{2x} - a(cy)^x \leqslant \frac{x}{4c^2y^2}\ a(2cy)^{2(x-1)}$, which imply the thesis of this lemma. □

Lemma 25. *Let $s \in \mathbb{Z}$, $x \in \mathbb{N}$ and $s \leqslant \lg\left(\frac{x}{4}\right)$. Then $a(2^{-s}x)^{2x+1} < \exp\left(-2^{s+1}\right)$ and*

$$a\left(2^{-s}x\right)^{x-1} > \exp\left(-2^s\right)\left(1 - \frac{2^{2s-1}-2^s}{x} - \frac{2^{2s-7}+2^{4s-3}}{x^2}\right)\ .$$

Proof. Consider a function defined on $(0,\infty)\times\mathbb{Z}$: $f_1(x;s) := a\left(2^{-s}x\right)^{2x+1} = \exp(-2^{s+1})\left(1 - \mathrm{O}\left(x^{-1}\right)\right)$. Realize a simple fact, that $z\ln(z) \geqslant z-1$ for $0 < z \leqslant 1$. Hence

$$\left(1-\frac{2^s}{x}\right)^{-2x}\frac{\partial f_1(x;s)}{\partial x} \underset{W}{=} \frac{2^s(2x+1)}{x^2} + 2\left(1-\frac{2^s}{x}\right)\ln\left(1-\frac{2^s}{x}\right) > \frac{2^{s-1}}{x^2} > 0$$

[2] See e.g. Faulhaber formula (1.13).

and in a consequence $a\left(2^{-s}x\right)^{2x+1} < \exp\left(-2^{s+1}\right)$ for any reasonable s. Moreover, let

$$D(x;s) := 1 - \frac{2^{2s-1}-2^s}{x} - \frac{2^{2s-7}+2^{4s-3}}{x^2}$$

and

$$f_2(x;s) := \frac{a\left(2^{-s}x\right)^{x-1}}{D(x;s)} \overset{W}{=} \exp(-2^s)\left(1+\mathrm{O}\left(x^{-2}\right)\right)^{3}$$

Then, in a similar way

$$\begin{aligned}
&D(x;s)^2\left(1-\frac{2^s}{x}\right)^{-x+1}\frac{\partial f_2(x;s)}{\partial x} \overset{W}{=}\\
&D(x;s)\left(\frac{2^s(x-1)}{x^2\left(1-\frac{2^s}{x}\right)}+\ln\left(1-\frac{2^s}{x}\right)\right)-\left(\frac{2^{2s-6}+2^{4s-2}}{x^3}+\frac{2^{2s-1}-2^s}{x^2}\right)\\
&<\left(\frac{2^{2s-1}-2^s}{x^2}+\frac{2^{3s}-2^{2s}}{x^3\left(1-\frac{2^s}{x}\right)}-\frac{2^{3s}}{3x^3}\right)-\left(\frac{2^{2s-6}+2^{4s-2}}{x^3}+\frac{2^{2s-1}-2^s}{x^2}\right)\\
&=\frac{2^{3s}-2^{2s}}{x^3\left(1-\frac{2^s}{x}\right)}-\frac{2^{3s}}{3x^3}-\frac{2^{2s-6}+2^{4s-2}}{x^3}\ .
\end{aligned}$$

Let $d := 1-\frac{2^s}{x}$ and realize that $d \in [\frac{3}{4},1)$ and $2^s-1+d\left(-\frac{2^s}{3}-2^{-6}-2^{2s-2}\right) > 0$. Indeed, if we put $z = 2^s$, then we attain a quadratic inequality in z variable, with determinant $\Delta = 1-\frac{5d}{3}+\frac{55d^2}{576}$, that is negative for $d \in [3/4,1)$.

Hence $\dfrac{\partial f_2(x;s)}{\partial x} < 0$ and consequently

$$a\left(2^{-s}x\right)^{x-1} > \exp\left(-2^s\right)\left(1-\frac{2^{2s-1}-2^s}{x}-\frac{2^{2s-7}+2^{4s-3}}{x^2}\right)$$

for any reasonable s. □

E.2.1 Formulation and proof of Lemma 26

Lemma 26. *The following two monotonic properties are satisfied:*

a) Sequence $(p_{2^k+1,k+4})_{k=2}^{\infty}$ is descending,

b) Sequence $(p_{2^k+1,k+5})_{k=3}^{\infty}$ is ascending.

Proof. Let $x = 2^k$ and $t \in \{0,1\}$. In advance we define

$$\kappa(k,t) := (-1)^{k+4+t}2^{-\frac{(k+4+t)(k+3+t)}{2}}r_{k+4+t}2^{-2x-1}$$

and

$$\tau(k,t) := [\![2 \not| (k+t)]\!](-1)^{k+t+3}2^{-\frac{(k+t+3)(k+t+2)}{2}}2r_{k+t+3}\left(\left(\frac{3}{4}\right)^{2x+1}-\left(\frac{1}{2}\right)^{x+2}\right)\ .$$

[3] D and f_2 are also defined on $(0,\infty)\times\mathbb{Z}$.

Realize that for $t \in \{0,1\}$ and $k \geqslant 5$, $|\tau(k,t)+\kappa(k,t)| < 2^{-50} < 10^{-15}$. Now, consider the differences between the consecutive elements of sequences:

$$
\begin{aligned}
&p_{2^{k+1}+1,k+5+t} - p_{2^k+1,k+4+t} \overset{\text{Thm } 4.5.2}{=} \kappa(k,t) + \\
&+ \sum_{i=0}^{k+3+t} (-1)^i 2^{-\frac{i(i-1)}{2}} r_i r_{k+t+4-i} \left[\left(1 - \frac{2^{-5-t+i}}{x}\right)^{2x+1} - \left(1 - \frac{2^{-4-t+i}}{x}\right)^{x+2} \right] \\
&= \sum_{i=0}^{\lfloor \frac{k+2+t}{2} \rfloor} \left[2^{-i(2i-1)} r_{2i} r_{k+t+4-2i} \left[a\left(2^{5+t-2i}x\right)^{2x+1} - a\left(2^{4+t-2i}x\right)^{x+2} \right] \right. \\
&\left. - 2^{-(2i+1)i} r_{2i+1} r_{k+t+3-2i} \left[a\left(2^{4+t-2i}x\right)^{2x+1} - a\left(2^{3+t-2i}x\right)^{x+2} \right] \right] + (\tau+\kappa)(k,t) \\
&= \sum_{i=0}^{\lfloor \frac{k+t+2}{2} \rfloor} 2^{-i(2i-1)} r_{2i+1} r_{k+t+4-2i} \left[a(2^{2i+1}) \left(a\left(2^{5+t-2i}x\right)^{2x+1} - a\left(2^{4+t-2i}x\right)^{x+2} \right) \right. \\
&\left. - 2^{-2i} a\left(2^{4+t-2i}x\right) \left(a\left(2^{4+t-2i}x\right)^{2x+1} - a\left(2^{3+t-2i}x\right)^{x+2} \right) \right] + \tau(k,t) + \kappa(k,t) \ .
\end{aligned}
$$

Let us define $u_t := 2^{5+t-2i}$ and

$$
\begin{aligned}
W_t(i) := a\left(2^{2i+1}\right) &\left(a\left(u_t x\right)^{2x+1} - a\left(\frac{u_t}{2}x\right)^{x+2} \right) \\
&- 2^{-2i} a\left(\frac{u_t}{2}x\right) \left(a\left(\frac{u_t}{2}x\right)^{2x+1} - a\left(\frac{u_t}{4}x\right)^{x+2} \right)
\end{aligned}
$$

and consider an upper bound of the last term:

$$
\begin{aligned}
W_t(i) &\leqslant a(2^{2i+1}) \left(a(u_t x)^{2x+1} - a(u_t x)^{2x+2} \left(a(u_t x)^2 - \frac{x+2}{u_t^2 x^2} \right) \right) \\
&\quad - 2^{-2i} \left(a\left(\frac{u_t}{4}x\right)^x \left(a\left(\frac{u_t}{4}x\right) + \frac{x+1}{\frac{u_t^2}{4}x^2} \right) - a\left(\frac{u_t}{2}x\right) a\left(\frac{u_t}{4}x\right)^{x+2} \right) \\
&\overset{W}{=} a(2^{2i+1}) a(u_t x)^{2x+1} \frac{1}{x} \left(\frac{3u_t+1}{u_t^2} - \frac{u_t+1}{u_t^3 x} - \frac{1}{u_t^3 x^2} \right) \\
&\quad - 2^{-2i} a\left(\frac{u_t}{4}x\right)^x \frac{1}{x} \left(\frac{6u_t+4}{u_t^2} - \frac{28}{x u_t^2} + \frac{32}{x^2 u_t^3} \right) \ .
\end{aligned}
$$

Note that $2i \leqslant k+2+t$, so $\frac{8}{x} \leqslant u_t$ and in consequence $6u_t - \frac{28}{x} > \frac{20}{x} > 0$. Moreover

$$
u_t(3u_t+1) - \frac{u_t+1}{x} - \frac{1}{x^2} \geqslant u_t \left(\frac{24}{x} + 1 \right) - \frac{u_t+1}{x} - \frac{1}{x^2} \geqslant \frac{7}{x} + \frac{183}{x^2} > 0 \ .
$$

Hence

$$
\begin{aligned}
&W_t(i) < a\left(2^{2i+1}\right) \exp\left(-\frac{2}{u_t}\right) \frac{1}{x} \left(\frac{3u_t+1}{u_t^2} - \frac{u_t+1}{x u_t^3} - \frac{1}{x^2 u_t^3} \right) \\
&- 2^{-2i} a\left(\frac{u}{4}x\right) \exp\left(-\frac{4}{u_t}\right) D(x; 2i-3-t) \frac{1}{x} \left(\frac{6u_t+4}{u_t^2} - \frac{28}{x u_t^2} + \frac{32}{x^2 u_t^3} \right) \overset{W}{=} \ldots
\end{aligned}
$$

$$\begin{aligned}
&\overset{W}{=} a\left(2^{2i+1}\right)\exp\left(-\frac{2}{u_t}\right)\frac{1}{x}\left(\frac{3u_t+1}{u_t^2}-\frac{u_t+1}{xu_t^3}-\frac{1}{x^2u_t^3}\right)\\
&-\frac{2^{-2i}}{x}\exp\left(-\frac{4}{u_t}\right)\left[\frac{6u_t+4}{u_t^2}-\frac{32+48u_t+28u_t^2}{u_t^4x}\right.\\
&-\frac{128+64u_t-\frac{703}{2}u_t^2+\frac{259}{4}u_t^3}{u_t^6x^2}+\frac{512+128u_t-1150u_t^2+\frac{909}{2}u_t^3}{u_t^7x^3}\\
&\left.-\frac{4608-1024u_t+530u_t^2}{u_t^7x^4}+\frac{4096+16u_t^2}{u_t^8x^5}\right]=:U_t(x;u_t(i))\ .
\end{aligned}$$

Analogically we would like to establish a lower bound of $W_t(i)$:

$$\begin{aligned}
W_t(i)\geqslant&a\left(2^{2i+1}\right)\left(a(u_tx)a\left(\frac{u_t}{2}x\right)^{x-1}\left(a\left(\frac{u_t}{2}x\right)+\frac{1}{u_t^2x}\right)-a\left(\frac{u_t}{2}x\right)^{x+2}\right)\\
&-2^{-2i}a\left(\frac{u_t}{2}x\right)\left(a\left(\frac{u_t}{2}x\right)^{2x+1}-a\left(\frac{u_t}{2}x\right)^{2x+2}\left(a\left(\frac{u_t}{2}x\right)^2-\frac{x+2}{\frac{u_t^2}{4}x^2}\right)\right)\\
\overset{W}{=}&a\left(2^{2i+1}\right)a\left(\frac{u_t}{2}x\right)^{x-1}\frac{1}{x}\left(\frac{3u_t+1}{u_t^2}-\frac{10u_t+1}{u_t^3x}+\frac{8}{u_t^3x^2}\right) \qquad \text{(E.1)}\\
&-2^{-2i}a\left(\frac{u_t}{2}x\right)^{2x+2}\frac{1}{x}\left(\frac{6u_t+4}{u_t^2}-\frac{4u_t+8}{u_t^3x}-\frac{8}{u_t^3x^2}\right)
\end{aligned}$$

Now from $\frac{8}{x}\leqslant u_t$ we attain

$$\begin{aligned}
u_t(3u_t+1)-\frac{10u_t+1}{x}&=u_t\left(\frac{7u_t}{4}+\frac{7}{8}\right)+(10u_t+1)\left(\frac{u_t}{8}-\frac{1}{x}\right)\\
&\geqslant u_t\left(\frac{7u_t}{4}+\frac{7}{8}\right)>0 \qquad \text{(E.2)}
\end{aligned}$$

and

$$u_t(6u_t+4)-\frac{4u_t+8}{x}-\frac{8}{x^2}\geqslant\frac{48u_t}{x}+\frac{32}{x}-\frac{4u_t+8}{x}-\frac{8}{x^2}\geqslant\frac{24}{x}+\frac{344}{x^2}>0\ .$$

Hence

$$\begin{aligned}
&W_t(i)>a\left(2^{2i+1}\right)\exp\left(-\frac{2}{u_t}\right)D(x;2i-4-t)\frac{1}{x}\left(\frac{3u_t+1}{u_t^2}-\frac{10u_t+1}{u_t^3x}+\frac{8}{u_t^3x^2}\right)\\
&-2^{-2i}\exp\left(-\frac{4}{u_t}\right)a\left(\frac{u_t}{2}x\right)\frac{1}{x}\left(\frac{6u_t+4}{u_t^2}-\frac{4u_t+8}{u_t^3x}-\frac{8}{u_t^3x^2}\right)\overset{W}{=}\dots
\end{aligned}$$

$$\begin{aligned}
&\overset{W}{=}\frac{a\left(2^{2i+1}\right)}{x}\exp\left(-\frac{2}{u_t}\right)\left[\frac{3u_t+1}{u_t^2}-\frac{2+5u_t+4u_t^2}{u_t^4x}-\frac{2+4u_t-\frac{575}{32}u_t^2+\frac{387}{32}u_t^3}{u_t^6x^2}\right.\\
&\left.+\frac{2+20u_t-\frac{511}{32}u_t^2+\frac{261}{16}u_t^3}{u_t^7x^3}-\frac{16+\frac{1}{4}u_t^2}{u_t^7x^4}\right]\\
&-2^{-2i}\exp\left(-\frac{4}{u_t}\right)\frac{1}{x}\left(\frac{6u_t+4}{u_t^2}-\frac{16u_t+16}{u_t^3x}+\frac{16}{u_t^4x^2}+\frac{16}{u_t^4x^3}\right)=:L_t(x;u_t(i))\ .
\end{aligned}$$

Now we show that $W_t(i) > 0$ for $i \geqslant 1$. Indeed, from inequalities (E.1) and (E.2) we obtain

$$W_t(i) > \frac{a\left(\frac{u_t}{2}x\right)^x}{xu_t^2}\left(a\left(2^{2i+1}\right)\frac{14u_t+7}{8} - 2^{-2i}(6u_t+4)\right) \tag{E.3}$$

If $i \geqslant 2$, then $E.3 \geqslant \frac{1}{256}\left(31(14u_t+7) - 96u_t - 64\right) > 0$.
In the last case, when $i = 1$, then $u_t \geqslant 8$, so

$$E.3 \geqslant \frac{1}{64}\left(7(14u_t+7) - 96u_t - 64\right) = \frac{2u_t - 15}{64} \geqslant \frac{1}{64}\ .$$

Thanks to the property $W_t(i) > 0$ for $i \geqslant 1$, we may subtly neutralize the influence of r_{k+5-i} in the considered sum:

$$\sum_{i=0}^{\lfloor\frac{k+2}{2}\rfloor} 2^{-i(2i-1)} r_{2i+1} r_{k+5-i} W_0(i) < r_{k+5} \sum_{i=0}^{\lfloor\frac{k+2}{2}\rfloor} 2^{-i(2i-1)} r_{2i+1} W_0(i)\ .$$

Naturally we may consider $U_0(x; u_0(i))$ instead of $W_0(i)$ numerically for $i \leqslant 4$:

$$\sum_{i=0}^{4} 2^{-i(2i-1)} r_{2i+1} U_0(x; u_0(i)) \overset{W}{=} -8.294491525704523 \cdot 10^{-6} + \frac{0.15588}{x} + \frac{0.00407163}{x^2} - \frac{0.0298032}{x^3} + \frac{0.0198815}{x^4} - \frac{0.00785419}{x^5}\ ,$$

so for $x \geqslant 2^{15}$ $(k \geqslant 15)$, $\sum\limits_{i=0}^{4} 2^{-i(2i-1)} r_{2i+1} W_0(i) \leqslant -3.53741 \cdot 10^{-6}$. Moreover we may bound $W_0(i)$ by $a(2^{5-2i}x)^{2x+1}$ for the rest of the sum:

$$\sum_{i=5}^{\lfloor\frac{k+2}{2}\rfloor} 2^{-i(2i-1)} r_{2i+1} a(2^{5-2i}x)^{2x+1} \leqslant \frac{R\ 2^{-45}e^{-64}}{1 - 2^{-21}e^{-192}} \overset{W}{=} 1.5784\ldots \cdot 10^{-41}\ ,$$

so $p_{2^{k+1}+1,k+5} - p_{2^k+1,k+4} < 0$ for $k \geqslant 15$.
However, according to Theorem 4.5.2, we also present the numerical values of the sequence $(p_{2^k+1,k+4})_{k=2}^{14}$ in Table E.1. We can now easily see that for any $k \geqslant 2$ we attained $p_{2^{k+1}+1,k+5} - p_{2^k+1,k+4} < 0$.

Moreover, realize that $r_{k+5}/r_{k+3} < 1.1$ for any $k \geqslant 3$, so

$$\sum_{i=0}^{1} 2^{-i(2i-1)} r_{2i+1} 1.1^{1-i} L_1(x; u_1(i)) \overset{W}{=} 0.00128843\ldots + \frac{0.00212699\ldots}{x} - \frac{0.00326251\ldots}{x^2} + \frac{0.000219133}{x^3} - \frac{3.50924875\ldots \cdot 10^{-7}}{x^4}$$

For any possible $x \geqslant 8$ $(k \geqslant 3)$, $\sum\limits_{i=0}^{1} 2^{-i(2i-1)} r_{2i+1} 1.1^{1-i} L_1(x; u_t(i)) > 0.0015$.
We already know that $W_1(i)$ are positive for $i > 1$, so $p_{2^{k+1}+1,k+6} - p_{2^k+1,k+5} > 0$ for all $k \geqslant 3$. □

k	$p_{2^k+1,k+4}$	k	$p_{2^k+1,k+4}$	k	$p_{2^k+1,k+4}$
2	$0.0000305176\ldots$	7	$0.0000189841\ldots$	12	$0.0000185484\ldots$
3	$0.0000256707\ldots$	8	$0.0000187590\ldots$	13	$0.0000185413\ldots$
4	$0.0000221583\ldots$	9	$0.0000186466\ldots$	14	$0.0000185378\ldots$
5	$0.0000203424\ldots$	10	$0.0000185904\ldots$		
6	$0.0000194356\ldots$	11	$0.0000185624\ldots$		

Table E.1: Numerical values of the sequence $(p_{2^k+1,k+4})_{k=2}^{14}$.

We may use Theorem 4.5.2 once again to see that $\frac{p_{2^6+1,10}}{p_{2^6+1,11}} = 129.454\ldots > 2^7$ and $\frac{p_{2^7+1,11}}{p_{2^7+1,12}} = 125.065\ldots < 2^7$. Together with Lemma 26 we may easily attain Claim 1 and we instantly see that it cannot be extended naturally for $k < 7$.

E.3 Final lemmas

Lemma 27. *Let $2 \leqslant l \leqslant n$ and assume that $p_{n,l-i} = 2^{2-i}\alpha_i p_{n,l-i+1}$ for $i \in [0:2]$ and $p_{n+1,l-j} = 2^{2-j}\alpha'_j p_{n+1,l-j+1}$ for $j \in [0:1]$.*
If $0 \leqslant \alpha_2 < \alpha_1 < \alpha_0$, then $0 < \alpha'_1 < \alpha'_0$.

Proof. Realize that $p_{n+1,l-i+1} = p_{n,l-i+1}(1 - 2^{-l+i-1} + 2^{-l+2}\alpha_i)$ for $i \in [0:2]$, so for $j \in [0:1]$,

$$\begin{aligned}\alpha'_j &= \frac{p_{n+1,l-j}}{2^{2-j}p_{n+1,l-j+1}} = \frac{p_{n,l-j}(1 - 2^{-l+j} + 2^{-l+2}\alpha_{j+1})}{2^{2-j}p_{n,l-j+1}(1 - 2^{-l+j-1} + 2^{-l+2}\alpha_j)} \\ &= \frac{\alpha_j(1 - 2^{-l+j} + 2^{-l+2}\alpha_{j+1})}{1 - 2^{-l+j-1} + 2^{-l+2}\alpha_j} \ .\end{aligned}$$

Assume that $\alpha'_1 \geqslant \alpha'_0$. Then

$$\mathfrak{L} := \alpha_1(1-2^{-l+1}+2^{-l+2}\alpha_2)(1-2^{-l-1}+2^{-l+2}\alpha_0) \geqslant \alpha_0(1-2^{-l}+2^{-l+2}\alpha_1)^2 =: \mathfrak{R}.$$

However, contrary to the assumption,

$$\begin{aligned}\mathfrak{L} &= \alpha_1(1 - 2^{-l+1} + 2^{-2l} - 2^{-l-1} + 2^{-l+2}(\alpha_0 + \alpha_2) - 2^{-2l+3}\alpha_0 \\ &\quad - 2^{-2l+1}\alpha_2 + 2^{-2l+4}\alpha_0\alpha_2) \\ &< \alpha_0(1 - 2^{-l+1} + 2^{-2l}) + \alpha_1(2^{-l+2}(2\alpha_0) + 2^{-2l+4}\alpha_0\alpha_1) \\ &< \alpha_0(1 - 2^{-l+1} + 2^{-2l} + \alpha_1(2^{-l+3} + 2^{-2l+3} + 2^{-2l+4}\alpha_1)) = \mathfrak{R} \ .\end{aligned}$$

□

Lemma 28. *If for some $n \in \mathbb{N}$,*
$p_{n,n} = 2^n\eta_n p_{n,n+1}$ and $p_{n+1,n+1} = 2^{n+1}\eta_{n+1}p_{n+1,n+2}$, then $\eta_n < \eta_{n+1}$.

Proof.

$$0 = p_{n+1,n+1} - 2^{n+1}\eta_{n+1}p_{n+1,n+2} = p_{n,n+1}(1 - 2^{-n-1}) + p_{n,n}2^{-n}$$
$$- \eta_{n+1}p_{n,n+1} = p_{n,n+1}(1 - 2^{-n-1} + \eta_n - \eta_{n+1}) \ ,$$

but $1 - 2^{-n-1} > 0$, so $\eta_n < \eta_{n+1}$. □

List of Tables

List of Figures

List of Algorithms

Bibliography

[1] Milton Abramowitz and Irene A. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Dover, New York, ninth dover printing, tenth gpo printing edition, 1964.

[2] Dan Alistarh, James Aspnes, and Rati Gelashvili. Space-optimal majority in population protocols. In *Proceedings of the 2018 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2221–2239, 2018.

[3] Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. *Journal of Computer and System Sciences*, 58(1):137 – 147, 1999.

[4] Tom M. Apostol. An elementary view of euler's summation formula. *The American Mathematical Monthly*, 106(5):409–418, 1999.

[5] László Babai. Monte-carlo algorithms in graph isomorphism testing, 1979.

[6] Brian Babcock, Shivnath Babu, Mayur Datar, Rajeev Motwani, and Jennifer Widom. Models and issues in data stream systems. In Lucian Popa, Serge Abiteboul, and Phokion G. Kolaitis, editors, *Proceedings of the Twenty-first ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 3-5, Madison, Wisconsin, USA*, pages 1–16. ACM, 2002.

[7] Brian Babcock, Mayur Datar, and Rajeev Motwani. Sampling from a moving window over streaming data. In David Eppstein, editor, *Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms, January 6-8, 2002, San Francisco, CA, USA*, pages 633–634. ACM/SIAM, 2002.

[8] Carlos Baquero, Paulo Sérgio Almeida, and Raquel Menezes. Fast estimation of aggregates in unstructured networks. In *ICAS*, pages 88–93, 2009.

[9] Carlos Baquero, Paulo Sérgio Almeida, Raquel Menezes, and Paulo Jesus. Extrema propagation: Fast distributed estimation of sums and network sizes. *IEEE Trans. Parallel Distrib. Syst.*, 23:668–675, 2012.

[10] Reuven Bar-Yehuda, Oded Goldreich, and Alon Itai. Efficient emulation of single-hop radio network with collision detection on multi-hop radio network with no collision detection. *Distrib. Comput.*, 5(2):67–71, September 1991.

[11] Petra Berenbrink, George Giakkoupis, and Peter Kling. Optimal Time and Space Leader Election in Population Protocols. In *STOC 2020 - 52nd Annual ACM Symposium on Theory of Computing*, pages 1–29, Chicago, United States, June 2020. ACM. Full verions, including all proofs.

[12] Jacob Bernoulli. Ars conjectandi. opus posthumum. `https://library.si.edu/digital-library/book/jacobibernoulli00bern`, 1713.

[13] Dimitri P. Bertsekas, editor. *Constrained Optimization and Lagrange Multiplier Methods*. Academic Press, 1982.

[14] Patrick Billingsley. *Probability and Measure*. Wiley Series in Probability and Statistics. Wiley, anniversary edition, 2012.

[15] Burton H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM*, 13(7):422–426, July 1970.

[16] Béla Bollobás. Random graphs. In *Modern Graph Theory*, pages 215–252. Springer, 1998.

[17] Vladimir Braverman, Rafail Ostrovsky, and Carlo Zaniolo. Optimal sampling from sliding windows. *J. Comput. Syst. Sci.*, 78(1):260–272, 2012.

[18] Karl Bringmann and Tobias Friedrich. Exact and efficient generation of geometric random variates and random graphs. In *Proceedings of the 40th International Conference on Automata, Languages, and Programming - Volume Part I*, ICALP'13, pages 267–278, Berlin, Heidelberg, 2013. Springer-Verlag.

[19] Seung Geol Choi, Dana Dachman-Soled, Mukul Kulkarni, and Arkady Yerukhimovich. Differentially-private multi-party sketching for large-scale statistics. *Proc. Priv. Enhancing Technol.*, 2020(3):153–174, 2020.

[20] Jacek Cichoń and Karol Gotfryd. Average counting via approximate histograms. *ACM Trans. Sen. Netw.*, 14(2), March 2018.

[21] Jacek Cichoń, Rafal Kapelko, and Dominik Markiewicz. On leader green election. In *Proceedings of the 27th International Conference on Probabilistic, Combinatorial and Asymptotic Methods for the Analysis of Algorithms, AofA'16*, pages 30–40, Kraków, Poland, July 2016.

[22] Jacek Cichoń and Marek Klonowski. On flooding in the presence of random faults. *Fundam. Inform.*, 123(3):273–287, 2013.

[23] Jacek Cichoń, Jakub Lemiesz, and Marcin Zawada. On message complexity of extrema propagation techniques. In *Proceedings of the 11th International Conference on Ad-Hoc, Mobile, and Wireless Networks*, ADHOC-NOW'12, page 1–13, Berlin, Heidelberg, 2012. Springer-Verlag.

[24] Jacek Cichon and Wojciech Macyna. Approximate counters for flash memory. In *2011 IEEE 17th International Conference on Embedded and Real-Time Computing Systems and Applications*, volume 1, pages 185–189. IEEE, 2011.

[25] E. Cohen. All-distances sketches, revisited: Hip estimators for massive graphs analysis. *IEEE Transactions on Knowledge and Data Engineering*, 27(9):2320–2334, 2015.

[26] R. M. Corless, G. H. Gonnet, D. E. G. Hare, D. J. Jeffrey, and D. E. Knuth. On the Lambert W function. *Adv. Comput. Math.*, 5(4):329–359, 1996.

[27] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein. *Introduction to Algorithms*. Computer science. McGraw-Hill, 2009.

[28] Alejandro Cornejo and Fabian Kuhn. Deploying wireless networks with beeps. In *Proceedings of the 24th International Conference on Distributed Computing*, DISC'10, pages 148–162, Berlin, Heidelberg, 2010. Springer-Verlag.

[29] Davide Crippa and Klaus Simon. Q-distributions and markov processes. *Discrete Math.*, 170(1):81–98, June 1997.

[30] Miklós Csűrös. Approximate counting with a floating-point counter. In *International Computing and Combinatorics Conference*, pages 358–367. Springer, 2010.

[31] Mayur Datar, Aristides Gionis, Piotr Indyk, and Rajeev Motwani. Maintaining stream statistics over sliding windows. *SIAM J. Comput.*, 31(6):1794–1813, 2002.

[32] Mayur Datar and Rajeev Motwani. The sliding-window computation model and results. In Minos N. Garofalakis, Johannes Gehrke, and Rajeev Rastogi, editors, *Data Stream Management - Processing High-Speed Data Streams*, Data-Centric Systems and Applications, pages 149–165. Springer, 2016.

[33] Damien Desfontaines, Andreas Lochbihler, and David Basin. Cardinality estimators do not preserve privacy. *Proceedings on Privacy Enhancing Technologies*, 2019(2):26–46, 2019.

[34] Dave Dice, Yossi Lev, and Mark Moir. Scalable statistics counters. In *Proceedings of the twenty-fifth annual ACM symposium on Parallelism in algorithms and architectures*, pages 43–52, 2013.

[35] Cynthia Dwork. Differential privacy. In *Automata, Languages and Programming, 33rd International Colloquium, ICALP 2006*, pages 1–12, 2006.

[36] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 486–503. Springer, 2006.

[37] Cynthia Dwork and Jing Lei. Differential privacy and robust statistics. In *STOC*, volume 9, pages 371–380, 2009.

[38] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, volume 3876, pages 265–284. Springer, 2006.

[39] Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N Rothblum. Differential privacy under continual observation. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 715–724. ACM, 2010.

[40] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014.

[41] G. Einziger, B. Fellman, R. Friedman, and Y. Kassner. Ice buckets: Improved counter estimation for network measurement. *IEEE/ACM Transactions on Networking*, 26(03):1165–1178, may 2018.

[42] Bennett Eisenberg. On the expectation of the maximum of iid geometric random variables. *Statistics & Probability Letters*, 78(2):135–143, 2008.

[43] James Allen Fill, Hosam M. Mahmoud, and Wojciech Szpankowski. On the distribution for the duration of a randomized leader election algorithm. *Ann. Appl. Probab*, 6:1260–1283, 1996.

[44] Philippe Flajolet. Approximate counting: a detailed analysis. *BIT Numerical Mathematics*, 25(1):113–134, 1985.

[45] Philippe Flajolet, Éric Fusy, Olivier Gandouet, and Frédéric Meunier. Hyperloglog: the analysis of a near-optimal cardinality estimation algorithm. In *AofA: Analysis of Algorithms*, pages 137–156. Discrete Mathematics and Theoretical Computer Science, 2007.

[46] Philippe Flajolet and G Nigel Martin. Probabilistic counting algorithms for data base applications. *Journal of computer and system sciences*, 31(2):182–209, 1985.

[47] Philippe Flajolet and Robert Sedgewick. *Analytic Combinatorics*. Cambridge University Press, USA, 1 edition, 2009.

[48] Alan Frieze and Michał Karonski. *Introduction to Random Graphs*. Cambridge University Press, 2015.

[49] Michael Fuchs, Chung-Kuei Lee, and Helmut Prodinger. Approximate Counting via the Poisson-Laplace-Mellin Method. In *23rd International Meeting on Probabilistic, Combinatorial, and Asymptotic Methods in the Analysis of Algorithms (AofA'12)*, pages 13–28. Discrete Mathematics and Theoretical Computer Science, 2012.

[50] R. G. Gallager, P. A. Humblet, and P. M. Spira. A distributed algorithm for minimum-weight spanning trees. *ACM Trans. Program. Lang. Syst.*, 5(1):66–77, January 1983.

[51] Mohsen Ghaffari and Bernhard Haeupler. Near optimal leader election in multi-hop radio networks. In *Proceedings of the Twenty-fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '13, pages 748–766, Philadelphia, PA, USA, 2013. Society for Industrial and Applied Mathematics.

[52] Leszek Gąsieniec, Grzegorz Stachowiak, and Przemyslaw Uznanski. Almost logarithmic-time space optimal leader election in population protocols. In *The 31st ACM Symposium on Parallelism in Algorithms and Architectures*, SPAA '19, page 93–102, New York, NY, USA, 2019. Association for Computing Machinery.

[53] R. William Gosper. Decision procedure for indefinite hypergeometric summation. *Proceedings of the National Academy of Sciences of the United States of America*, 75(1):40–42, 1978.

[54] Ronald L. Graham, Donald E. Knuth, and Oren Patashnik. *Concrete mathematics - a foundation for computer science (2. ed.)*. Addison-Wesley, 1994.

[55] André Gronemeier and Martin Sauerhoff. Applying approximate counting for computing the frequency moments of long data streams. *Theory of Computing Systems*, 44(3):332–348, 2009.

[56] Peter J. Haas. Data-stream sampling: Basic techniques and results. In *Data Stream Management*, 2016.

[57] Magnús Halldórsson, Stephan Holzer, and Evangelia Markatou. Leader election in sinr model with arbitrary power control. *Theoretical Computer Science*, 811, 01 2019.

[58] Stefan Heule, Marc Nunkesser, and Alexander Hall. Hyperloglog in practice: Algorithmic engineering of a state of the art cardinality estimation algorithm. In *Proceedings of the 16th International Conference on Extending Database Technology*, pages 683–692, Genoa, Italy, 2013.

[59] Changhui Hu, Jin Li, Zheli Liu, Xiaojie Guo, Yu Wei, Xuan Guang, Grigorios Loukides, and Changyu Dong. How to make private distributed cardinality estimation practical, and get differential privacy for free. In Michael Bailey and Rachel Greenstadt, editors, *30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021*, pages 965–982. USENIX Association, 2021.

[60] Wolfram Research, Inc. Mathematica, Version 11.3.

[61] Ozlem Durmaz Incel. A survey on multi-channel communication in wireless sensor networks. *Computer Networks*, 55(13):3081–3099, 2011.

[62] P. Indyk and D. Woodruff. Tight lower bounds for the distinct elements problem. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 283– 288, 11 2003.

[63] Piotr Indyk. Stable distributions, pseudorandom generators, embeddings and data stream computation. In *41st Annual Symposium on Foundations of Computer Science, FOCS 2000, 12-14 November 2000, Redondo Beach, California, USA*, pages 189–197. IEEE Computer Society, 2000.

[64] Philippe Jacquet, Dimitris Milioris, and Paul Mühlethaler. A novel energy efficient broadcast leader election. In *MASCOTS*, pages 495–504. IEEE, 2013.

[65] Svante Janson and Wojciech Szpankowski. Analysis of an asymmetric leader election algorithm. *Electr. J. Comb.*, 4(1), 1997.

[66] Witold Jarczyk and Miklos Laczkovich. Convexity on abelian groups. *Journal of Convex Analysis*, 16(1):33–48, 2009.

[67] E.I. Jury. *Theory and Applications of the Z-Transform Method*. New York, 1964.

[68] Peter Kirschenhofer and Helmut Prodinger. The number of winners in a discrete geometrically distributed sample. *The Annals of Applied Probability*, 6(2):687–694, May 1996.

[69] M.S. Klamkin and D.J. Newman. Extensions of the weierstrass product inequalities. *Mathematics Magazine*, 43(3):137–141, 01 1970.

[70] Donald E. Knuth. Mathematics and computer science: Coping with finiteness. *Science*, 194(4271):1235–1242, 1976.

[71] Donald E. Knuth. Johann Faulhaber and sums of powers. *Mathematics of Computation*, 61(203):277–294, 1993.

[72] Gérard Le Lann. Distributed systems-towards a formal approach. In *IFIP congress*, volume 7, pages 155–160. Toronto, 1977.

[73] D. H. Lehmer. On the maxima and minima of bernoulli polynomials. *The American Mathematical Monthly*, 47(8):533–538, 1940.

[74] Kim-Hung Li. Reservoir-sampling algorithms of time complexity ¡i¿o¡/i¿(¡i¿n¡/i¿(1 + log(¡i¿n¡/i¿/¡i¿n¡/i¿))). *ACM Trans. Math. Softw.*, 20(4):481–493, December 1994.

[75] Jens Lienig and Hans Bruemmer. *Fundamentals of Electronic Systems Design*. Springer Publishing Company, Incorporated, 1st edition, 2017.

[76] G. Louchard and H. Prodinger. The asymmetric leader election algorithm: Another approach. *Annals of Combinatorics*, 12:449–478, 2009.

[77] S.M. Masum, A.A. Ali, and M.T.-yI. Bhuiyan. Asynchronous leader election in mobile ad hoc networks. In *20th International Conference on Advanced Information Networking and Applications - Volume 1 (AINA'06)*, volume 2, pages 5 pp.–, 2006.

[78] Robert M. Metcalfe and David R. Boggs. Ethernet: Distributed packet switching for local computer networks. *Commun. ACM*, 19(7):395–404, July 1976.

[79] Y. Métivier, J.M. Robson, and A. Zemmari. Analysis of fully distributed splitting and naming probabilistic procedures and applications. *Theoretical Computer Science*, 584:115 – 130, 2015. Special Issue on Structural Information and Communication Complexity.

[80] Robert Morris. Counting large numbers of events in small registers. *Communications of the ACM*, 21(10):840–842, 1978.

[81] Koji Nakano and Stephan Olariu. Randomized o(log log n)-round leader election protocols in packet radio networks. In Kyung-Yong Chwa and Oscar H. Ibarra, editors, *Algorithms and Computation*, pages 210–219, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

[82] Koji Nakano and Stephan Olariu. Randomized leader election protocols in radio networks with no collision detection. In Gerhard Goos, Juris Hartmanis, Jan van Leeuwen, D. T. Lee, and Shang-Hua Teng, editors, *Algorithms and Computation*, pages 362–373, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.

[83] Koji Nakano and Stephan Olariu. Uniform leader election protocols for radio networks. *Parallel and Distributed Systems, IEEE Transactions on*, 13:516 – 526, 06 2002.

[84] Arvind Narayanan and Vitaly Shmatikov. De-anonymizing social networks. In *Security and Privacy, 2009 30th IEEE Symposium on*, pages 173–187. IEEE, 2009.

[85] Arvind Narayanan and Vitaly Shmatikov. Myths and fallacies of personally identifiable information. *Communications of the ACM*, 53(6):24–26, 2010.

[86] J. Neyman. Outline of a theory of statistical estimation based on the classical theory of probability. *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 236(767):333–380, 1937.

[87] James Ritchie Norris. *Markov Chains*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 1997.

[88] Christopher R. Palmer, Phillip B. Gibbons, and Christos Faloutsos. Anf: A fast and scalable tool for data mining in massive graphs. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, page 81–90, New York, NY, USA, 2002. Association for Computing Machinery.

[89] Helmut Prodinger. How to select a loser. *Discrete Mathematics*, 120(1-3):149–159, 1993.

[90] quandl. Bitcoin mining statistics. `https://www.quandl.com/data/BITCOINWATCH/MINING`, 2017.

[91] Nicholas Riley and Craig Zilles. Probabilistic counter updates for predictor hysteresis and bias. *IEEE Computer Architecture Letters*, 5(1):18–21, 2006.

[92] Sbiis Saibian. One to Infinity: A Guide to the Finite. From Google Sites, 2008-2021. Sbiis Saibian's Large Number Site: `https://sites.google.com/site/largenumbers/home/appendix/a/numbers/265536`.

[93] Nicola Santoro. *Design and Analysis of Distributed Algorithms (Wiley Series on Parallel and Distributed Computing)*. Wiley-Interscience, USA, 2006.

[94] Robert Sedgewick. Cardinality estimation, 2018.

[95] Adam D. Smith, Shuang Song, and Abhradeep Thakurta. The flajolet-martin sketch itself preserves differential privacy: Private counting with minimal space. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[96] Stevo Stević. Asymptotic behaviour of a sequence defined by iteration. *Mat. Vesnik*, 48(03-04):99 – 105, 1996.

[97] Yuichi Sudo and Toshimitsu Masuzawa. Leader election requires logarithmic time in population protocols. *CoRR*, abs/1906.11121, 2019.

[98] S. Joshua Swamidass and Pierre Baldi. Mathematical correction for fingerprint similarity measures to improve chemical retrieval. *J. Chem. Inf. Model.*, 47(3):952–964, 2007.

[99] Wojciech Szpankowski and Vernon Rego. Yet another application of a binomial recurrence. Order statistics. *Computing*, 43(4):401–410, February 1990.

[100] Y. Tillé. *Sampling Algorithms*. Springer Series in Statistics. Springer, 2006.

[101] Daniel Ting. Streamed approximate counting of distinct elements: Beating optimal batch methods. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, page 442–451, New York, NY, USA, 2014. Association for Computing Machinery.

[102] Daniel Ting. Approximate distinct counts for billions of datasets. In *Proceedings of the 2019 International Conference on Management of Data*, SIGMOD '19, page 69–86, New York, NY, USA, 2019. Association for Computing Machinery.

[103] László Tóth. Transcendental infinite products associated with the ± 1 thue-morse sequence, 2020.

[104] Benjamin Van Durme and Ashwin Lall. Probabilistic counting with randomized storage. In *Twenty-First International Joint Conference on Artificial Intelligence*, 2009.

[105] Jeffrey Scott Vitter. Random sampling with a reservoir. *ACM Transactions on Mathematical Software*, 11(1):37–57, March 1985.

[106] Herbert S. Wilf and Doron Zeilberger. An algorithmic proof theory for hypergeometric (ordinary and "q") multisum/integral identities. *Inventiones mathematicae*, 108(1):575–633, Dec 1992.

[107] Dan E Willard. Log-logarithmic selection resolution protocols in a multiple access channel. *SIAM Journal on Computing*, 15(2):468–477, 1986.

[108] Wolfram Research, Inc., `http://functions.wolfram.com/06.03.06.0005.01`. Binomial coefficient, Expansions at k==infinity.

[109] X. Yun, G. Wu, G. Zhang, K. Li, and S. Wang. Fastraq: A fast approach to range-aggregate queries in big data environments. *IEEE Transactions on Cloud Computing*, 3(2):206–218, 2015.

[110] Doron Zeilberger. The method of creative telescoping. *Journal of Symbolic Computation*, 11(3):195 – 204, 1991.